

# BACnet<sup>®</sup> TESTING LABORATORIES

# INTERIM TEST SPECIFICATION

To Be Used with Test Package 15.1 Version 10 July 19, 2018

Approved by the BTL Working Group on July 12, 2018 Approved by the BTL Working Group Voting Members on August 8, 2018 Published on August 9, 2018

BACnet is a registered trademark of ASHRAE. ASHRAE does not endorse, approve or test products for compliance with ASHRAE standards. Compliance of listed products to the requirements of ASHRAE Standard 135 is the responsibility of BACnet International (BI). BTL is a registered trademark of BI.

#### Foreward

The purpose of this document is to define interim tests and other test package changes made to support testing of a device that supports functionality currently not covered in the released BTL Test Package. This document should be applied and used with BTL Test Package 15.

Vendors who are planning to submit a device for testing and who implement Protocol\_Revision 16 and higher, or which contain functionality not covered by the Official Test Package, should use this Interim Test document.

Please note that there may be other tests for other functional areas that may also be required for your device. Please contact the BTL Manager before submitting your device for testing to ensure you are aware of all tests that will need to be applied to your device.

The changes in this document are for interim use only and may or may not be used as documented here when the final changes are applied to the next Test Package revision. Devices tested using this interim test document shall be recalled for updated testing when the next revision of test package is released that includes the topics covered here.

The changes in this document are summarized below:

**BTL-TP15.0-0.1.0** Tests for the Network Port object (Protocol\_Revision 17 or higher) BTL-TP15.0-0.2.0 Tests for the Elevator Group object (Protocol\_Revision 18 or higher) BTL-TP15.0-0.3.0 Tests for the Escalator object (Protocol Revision 18 or higher) BTL-TP15.0-0.4.0 Tests for the Lift object (Protocol Revision 18 or higher) BTL-TP15.0-0.5.0 Network Port OPTIONAL properties clarified (Protocol Revision 18 or higher) BTL-TP15.0-0.6.0 Test of Write-BDT-NAK to Write-BDT service (Protocol\_Revision 17 or higher) BTL-TP15.0-0.7.0 Tests for the claim of NM-BBMDC-B (Protocol Revision 18 or higher) BTL-TP15.0-1.1.0 Tests for the FAULT\_LISTED algorithm (Protocol\_Revision 18 or higher) BTL-TP15.0-1.2.0 Tests for FAULT-to-FAULT transitions in FAULT\_LISTED algorithm (Protocol\_Revision 18 or higher) BTL-TP15.0-2.1.0 Binary Lighting Output object (Protocol\_Revision 16 or higher) BTL-TP15.0-3.1.0 NM-CE-A Test Considerations (Protocol Revision 2 or higher) BTL-TP15.0-4.1.0 Read-only Recipient List Test Considerations (Protocol Revision 13 or higher) BTL-TP15.0-4.2.0 Tests for the claim of AE-CRL-B (Protocol\_Revision 2 or higher) **BTL-TP15.0-5.1.0** Tests for the Lighting Output object (Protocol Revision 14 or higher) BTL-TP15.0-6.1.0 Tests for the claim of DS-COVP-B (Protocol Revision 2 or higher) BTL-TP15.0-7.1.0 Tests for the claim of NM-FDR-A (Protocol\_Revision 2 or higher) BTL-TP15.0-8.1.0 Tests for the claim of GW-EO-B (Protocol Revision 2 or higher) BTL-TP15.0-9.1.0 Tests for the Life Safety Point object (Protocol Revision 2 or higher) BTL-TP15.0-9.2.0 Tests for the Life Safety Zone object (Protocol\_Revision 2 or higher) BTL-TP15.0-9.3.0 Tests for the claim of AE-LS-A (Protocol Revision 2 or higher) BTL-TP15.0-9.4.0 Tests for the claim of AE-LS-B (Protocol\_Revision 2 or higher) **BTL-TP15.1-0.1.0** File object (Protocol\_Revision 2 or higher)

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135.1-2013 or any part of the Test Package 15.1 are indicated through the use of *italics*, while deletions are indicated by strikethrough. Where entirely new sections are proposed to be added, plain type is used throughout.

BTL-TP15.0-0.1.0 TESTS FOR THE NETWORK PORT OBJECT	5
3.X43 Network Port Object	5
3.X43.1 Base Requirements	5
3.X43.2 Supports writable Out_Of_Service properties	5
BTL-TP15.0-0.2.0 TESTS FOR THE ELEVATOR GROUP OBJECT	9
3.X45 Elevator Group Object	9
3.X45.1 Base Requirements	9
3.X45.2 Supports Group_Members Property	9
3.X45.3 Supports Landing_Call_Control Property	.10
BTL-TP15.0-0.3.0 TESTS FOR THE ESCALATOR OBJECT	.12
3.X46 Escalator Object	.12
3.X46.1 Base Requirements	.12
3.X46.2 Supports writable Out_Of_Service properties	.12
3.X46.3 Supports Escalator_Mode Property	.13
3.X46.4 Supports Energy_Meter_Ref Property	.13
3.X46.5 Supports CHANGE_OF_STATE event algorithm with Passenger_Alarm property	.14
3.X46.6 Supports Reliability_Evaluation_Inhibit Property	.14
BTL-TP15.0-0.4.0 TESTS FOR THE LIFT OBJECT	.20
3 X47 Lift Object	20
3 X47.1 Base Requirements	.20
3.X47.2 Supports writable Out Of Service properties	.20
3.X47.3 Supports Making_Car_Call and Register_Car_Call Properties	.22
3.X47.4 Supports BACnetARRAY Properties related to the doors of a car	.22
3.X47.5 Supports Landing_Door_Status and Car_Door_Status Properties	.23
3.X47.6 Supports Car_Position and Next_Stopping_Floor Properties	.23
3.X47.7 Supports Assigned_Landing_Calls, Making_Car_Call and Registered_Car_Call Properties	.23
3.X47.9 Supports Higher_Deck and Lower_Deck Properties	.24
3.X47.10 Supports Reliability_Evaluation_Inhibit Property	.24
3.X47.11 Supports Reliability Evaluation	.24
3.X47.12 Supports CHANGE_OF_STATE event algorithm with Passenger_Alarm property	.25
PTL TR15 0.05 0 TEST CONSIDER A TIONS FOR NETWORK BORT OPTIONAL PROPERTIES	.23
CLARIFIED	.37
4.4 Data Sharing - ReadPropertyMultiple - B	37
4.4.1 Base Requirements	.37
BTL-TP15.0-0.6.0 TEST OF WRITE-BDT-NAK TO WRITE-BDT SERVICE	.39
0.4 RACnet/IP Anney I RRMD	30
9.4 BACICUT - Amicx J - DBMD	.39
BTL-TP15.0-0.7.0 TEST CONSIDERATIONS FOR THE NM-BBMDC-B BIBB	.40
10 X3 Network Management - BACnet Broadcast Management Device Configuration - B	40
10.X3.1 Base Requirements	.40
10.X3.2 Supports Registration by Foreign Devices	.42
10.X3.3 Executes Write-Broadcast-Distribution-Table	.43
10.X3.4 Supports BBMD_Broadcast_Distribution_Table property	.44
BTL-TP15.0-1.1.0 TESTS FOR THE FAULT LISTED ALGORITHM	.47
BTL-TP15.0-1.2.0 TESTS FOR FAULT-TO-FAULT TRANSITIONS IN FAULT LISTED ALGORITHM	49

## **Table of Contents**

3.X46	5 Escalator Object	49
3.X	46.7 Supports FAULT-to-FAULT transitions in FAULT_LISTED	49
BTL-	TP15.0-2.1.0: BINARY LIGHTING OUTPUT OBJECT	52
3.X41 3.X	Binary Lighting Output Object	
3.X	41.2 Supports Command Prioritization	
3.X	(41.3 Supports Writable Out_Of_Service Properties	53
3.X	(41.4 Supports Blink-warn	53 55
3 X	(41.5 Supports writable Polarity property	
3.X	41.7 Supports Elapsed Active Time Tracking	
3.X	41.8 Contains an object with Reliability_Evaluation_Inhibit Property	55
BTL-	TP15.0-3.1.0 NM-CE-A TEST CONSIDERATIONS	65
10.X4	Vetwork Management - Connection Establishment - A	65
10.	X4.1 Base Requirements	65
BTL-	TP15.0-4.1.0 READ-ONLY RECIPIENT_LIST TEST CONSIDERATIONS	66
3.17 N	Notification Class Object	
· · · 3 1′	7.4 Supports read only Recipient List Properties	66 66
5.2 Al	larm and Event - Notification - Internal-B	
5.2	.1 Base Requirements	66
BTL-	TP15.0-5.1.0 TESTS FOR THE LIGHTING OUTPUT OBJECT	70
3.X54	Lighting Output Object	71
3.X	54.1 Base Requirements	
3.X	54.2 Supports Command Prioritization	71
3 X	54.5 Supports writable Out Of Service Properties	72 73
3.X	(54.5     Supports blink-warn	73
3.X	54.6 Supports Transition property	75
3.X	Supports Feedback_Value property	76
3.X	54.8 Supports Min_Actual_Value and Max_Actual_Value properties	76
3.X 4.1	0.X54 Supports COV for Lighting Output Objects	
BTL-	TP15.0-6.1.0 TESTS FOR THE CLAIM OF DS-COVP-B	
4 DA'	TA SHARING	
4.19 I	Data Sharing - Change Of Value Property - B	
4.1	9.1 Base Requirements	
BTL-	TP15.0-7.1.0 TESTS FOR THE CLAIM OF NM-FDR-A	100
10	NETWORK MANAGEMENT	100
10	NETWORK MANAGEMENT	100
10.X2	2 NETWORK MANAGEMENT - FOREIGN DEVICE REGISTRATION - A	100
10.2	X2.1 Base Requirements	100
10.	X2.2 Supports configurable BBMD Address	
10.	X2.3 Supports a mode where it transmits a Broadcast at Startup	
10.		102
BTL-	TP15.0-8.1.0 TESTS FOR THE CLAIM OF GW-EO-B	

<ul> <li>11 Gateway</li></ul>	
11.2.2 Supports writes that affect values in "gatewayed" devices	106
BTL-TP15.0-9.1.0: LIFE SAFETY POINT OBJECT	110
3.X50 Life Safety Point Object         3.X50.1 Base Requirements	110 110
BTL-TP15.0-9.2.0: LIFE SAFETY ZONE OBJECT	111
<ul> <li>3.X51 Life Safety Zone Object</li></ul>	
BTL-TP15.0-9.3.0 TESTS FOR THE CLAIM OF AE-LS-A	114
<ul> <li>5.22 Alarm and Event Management - Life Safety - A</li></ul>	
5 ALARM AND EVENT MANAGEMENT BIBBS	
<ul> <li>5.23 Alarm and Event Management - Life Safety - B</li></ul>	119 119 122 122 123 123 123 123 123 124 124 124 124 124
BTL-TP15.1-0.1.0 FILE OBJECT	128

## **BTL-TP15.0-0.1.0** Tests for the Network Port object

A device including a Network Port object must claim Protocol\_Revision 17 or higher and comply with the following section.

[In BTL Checklist, add new Network Port section in existing 3. Object testing.]

Support	Listing	Option	
Network Port Object			
	R	Base Requirements	
	S	Supports writable Out_Of_Service properties	

[In BTL Test Plan, add new Network Port section to 3. Object testing]

### 3.X43 Network Port Object

### **3.X43.1 Base Requirements**

Base requirements must be met by any IUT that can contain Network Port objects.

BTL - 7.3.2.X43.1 - Network Port ACTIVATE_CHANGES test		
Test Method Manual		Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X43.2 - Network	Port non-volatility properties test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	
BTL	- 9.18.X5 - ReadPropert	y of the Network Port Object using the Unknown Instance
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

### 3.X43.2 Supports writable Out\_Of\_Service properties

The Out\_Of\_Service property in Network Port objects contained in the IUT is either writable or can be modified by any other means.

BTL - 7.3.2.X43.3 - Out_Of_Service, Status_Flags, and Reliability test for an Object that does not		
contain Present_Value		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.

Test Conditionality	If this property is writable, this test must be executed.
<b>Test Directives</b>	This test shall be applied to a Network Port object.
<b>Testing Hints</b>	
Notes & Results	

[In BTL Specified Tests, add three new tests 7.3.2.X43.X1 through 7.3.2.X43.X3, and one ReadProperty positive service test 9.18.1.X5 as indicated.]

### 7.3.2.X43.1 Network Port ACTIVATE\_CHANGES test

Reason for Change: New test per Addendum 135-2012ai.

Purpose: This test verifies that after any of the Network Port properties are changed, the revised value is activated when executing a ReinitializeDevice ACTIVATE\_CHANGES service request.

Test Concept: Write any of the writable properties of a Network Port object, and activate those changes by issuing a ReinitializeDevice – WARMSTART or ACTIVATE\_CHANGES service request. Then after the IUT has time to have finished its update, verify that the Network Port object properties contain the values that were written.

Test Steps:

- 1. WRITE (any writable Network Port property) = (a value different from current value)
- 2. VERIFY Changes\_Pending = TRUE
- TRANSMIT ReinitializeDevice-Request
   'Reinitialized State of Device' = WARMSTART | ACTIVATE\_CHANGES
   'Password' = (any valid password)
- 4. RECEIVE BACnet-SimpleACK-PDU
- 5. CHECK (that the IUT has had time to have finished its update)
- 6. REPEAT X for each changed Network Port property)
  - VERIFY X = (the revised value to which it was changed)
- 7. VERIFY Changes\_Pending = FALSE

#### 7.3.2.X43.2 Network Port non-volatility properties test

Reason for Change: New test per Addendum 135-2012ai.

Purpose: This test verifies that after any of the Network Port properties is changed, and the revised value is activated, then the revised value with which it was configured is maintained through a power failure and device restart.

Test Concept: Write any of the writable properties of a Network Port object (multiple properties may be written), and activate those changes by issuing a ReinitializeDevice – WARMSTART or ACTIVATE\_CHANGES service request. Then after the IUT has time to have finished its update, restart the IUT device by temporarily removing power. When the device has resumed operation after that restart, verify that the Network Port object properties contain the values that were changed and activated.

Test Steps:

- 1. WRITE (X, any writable Network Port property) = (a value different from current value)
- 2. TRANSMIT ReinitializeDevice-Request

### 'Reinitialized State of Device' = WARMSTART | ACTIVATE\_CHANGES 'Password' = (any valid password)

- 3. RECEIVE BACnet-SimpleACK-PDU
- 4. WAIT for IUT to have finished its update
- 5. CHECK (that the IUT has had time to have finished its update)
- 6. VERIFY X = (the revised value to which it was changed)
- 7. MAKE (the IUT power cycle to reinitialize)

8. VERIFY X = (the revised value to which it was changed)

# 7.3.2.X43.3 Out\_Of\_Service, Status\_Flags, and Reliability test for an Object that does not contain Present\_Value

Purpose: This test verifies the interrelationship between the Out\_Of\_Service, Status\_Flags, and Reliability properties. If the PICS indicates that the Out\_Of\_Service property of the object under test is not writable, and if the value of the property cannot be changed by other means, then this test shall be omitted. This test applies to objects that do not contain Present\_Value.

Test Concept: Write to and verify the interrelationship between the Out\_Of\_Service, Status\_Flags, and Reliability properties of an object which does not contain Present\_Value.

Configuration Requirements: The selected object is configured such that its Reliability is NO\_FAULT\_DETECTED before execution of this test.

Test Steps:

1. IF (Out\_Of\_Service is writable) THEN

WRITE Out\_Of\_Service = TRUE

ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, FALSE, ?, TRUE)
- 4. IF (Reliability is present and writable) THEN

REPEAT X = (all values of the Reliability enumeration appropriate to the object type except NO\_FAULT\_DETECTED) DO { WRITE Reliability = X VERIFY Reliability = X

VERIFY Status\_Flags = (TRUE, TRUE,?, TRUE) WRITE Reliability = NO\_FAULT\_DETECTED VERIFY Reliability = NO\_FAULT\_DETECTED VERIFY Status\_Flags = (? FALSE, ?, TRUE)

- 5. CHECK (all communication of the protocol modeled by the object, through that port is disabled)
- 6. IF (Out\_Of\_Service is writable) THEN

WRITE Out\_Of\_Service = FALSE

ELSE

}

MAKE (Out\_Of\_Service = FALSE)

- 7. VERIFY Out\_Of\_Service = FALSE
- 8. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 9.18.1.X5 ReadProperty of the Network Port Object using the Unknown Instance

Purpose: Verify that the IUT selects the correct object when it receives Network Port with special object instance of 4194303.

Test Concept: Execute a Read service request specifying 'Object Identifier' = (Network Port, 4194303). The responding BACnet-user shall treat the Object Identifier as if it correctly matched the local Network Port object representing the network port through which the request was received.

Configuration Requirements: Let X be the instance numbers of Network Port object (can be same or different objects) for the IUT. If the Protocol\_Revision claimed is less than 17, this test shall be skipped.

1.	TRANSMIT ReadProperty-R	lequest,	
	'Object Identifier' =	(Network Port, 4194303),	
	'Property Identifier' =	Object-Identifier	
2.	<b>RECEIVE</b> ReadProperty-AC	K,	
	'Object Identifier' =	(Network Port, X),	
	'Property Identifier' =	Object-Identifier,	
	'Property Value' =	(Network Port, X)	
3.	TRANSMIT ReadProperty-Request through the same port as above,		
	'Object Identifier' =	(Network Port, 4194303),	
	'Property Identifier' =	(P: any valid property which is present in the same local Network Port object as	
	above)		
4.	<b>RECEIVE</b> ReadProperty-AC	Κ,	
	'Object Identifier' =	(Network Port, X),	
	'Property Identifier' =	Р,	
	Property Value' -	(value of P from the local Network Port object representing the network port	

'Property Value' = (value of P from the local Network Port object representing the network port through which the request was received)

Passing Result: The IUT shall respond as indicated conveying the value from a local Network Port object representing the network port through which the request was received.

## **BTL-TP15.0-0.2.0** Tests for the Elevator Group object

A device including an Elevator Group object must claim Protocol\_Revision 18 or higher and comply with the following section.

[In BTL Checklist, add new Elevator Group section in existing 3.]

Support	Listing	Option	
Elevator Group			
	R	Base Requirements	
	R	Supports Group_Members property	
	O Supports Landing_Call_Control property		

[In BTL Test Plan, add new Elevator Group section at end of existing 3. Object testing, with sections 3.X45.1 Base Requirements, and two other 3.X45.2 through 3.X45.3 sections as indicated.]

### 3.X45 Elevator Group Object

### **3.X45.1 Base Requirements**

The object contains Machine\_Room\_ID Property.

BTL - '	TL - 7.3.2.X45.1.1 - Machine_Room_ID property linking with the Positive_Integer_Value Object		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X45.2 Supports Group\_Members Property

The object contains a Group\_Members Property.

BTL - 7.3.2.X45.1.2 - Linking of Lift Objects under Group\_Members property of the Elevator Group Object

Test Method	Manual
Configuration	As per BTL Specified Tests.
<b>Test Conditionality</b>	Must be executed if IUT supports Lift object.
Test Directives	
Testing Hints	
Notes & Results	

BTL - 7.3.2.X45.1.3 - Linking of Escalator Objects under Group\_Members property of the Elevator Group Object

-	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed if IUT supports Escalator object.
	Test Directives	
	Testing Hints	

Notes & Results	
-----------------	--

### 3.X45.3 Supports Landing\_Call\_Control Property

The object contains a Landing\_Call\_Control Property.

BTL - 7.3.2.X45.1.4 - Linking of Landing_Call_Control Property Test		
Test Method Manual		Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

[Add in BTL Specified Tests, these four new tests]

### 7.3.2.X45.1.1 Machine\_Room\_ID property linking with the Positive\_Integer\_Value Object

Purpose: To verify that Machine\_Room\_ID property of Elevator Group reference the Positive\_Integer\_Value (PIV) object, whose Present\_Value property contains the identification number for the machine room that contains the group of Lifts or Escalators, represented by this object.

Test Concept: A machine room contains the Elevator Group which is having a group of Lifts or Escalators. This machine room is mapped to the Present\_Value property of Positive\_Integer\_Value Object which in turn is referenced to the Machine\_Room\_ID property of Elevator Group.

Configuration Requirements: The Machine room contains Elevator Group (EG1). OBJECT is any valid object type. X is any valid instance number in the range 0 to 4194302.

Test Steps:

1. IF (Machine\_Room\_ID contains room identification number) THEN VERIFY (EG1), Machine\_Room\_ID = (PIV, X)

ELSE

VERIFY (EG1), Machine\_Room\_ID = (OBJECT, 4194303)

#### 7.3.2.X45.1.2 Linking of Lift Objects under Group\_Members property of the Elevator Group Object

Purpose: This test verifies that the Group\_Members property of the Elevator Group object contains the object identifier of the Lift object representing lifts contained within the group represented by this Elevator Group object.

Test Concept: Tester selects an Elevator Group and reads the Group\_Members property of the Elevator Group and verifies that all the Lifts that are configured under one group are present under the Group\_Members property of the Elevator Group object.

Configuration Requirements: Configure 2 Lifts (L1 and L2) under the Elevator Group (EG1).

Test Steps:

1. VERIFY (EG1), Group\_Members = (L1, L2)

#### 7.3.2.X45.1.3 Linking of Escalator Objects under Group\_Members property of the Elevator Group Object

Purpose: This test verifies that the Group\_Members property of the Elevator Group object contains the object identifier of the Escalator object representing the escalators contained within the group represented by this Elevator Group object.

Test Concept: Tester selects an Elevator Group and reads the Group\_Members property of the Elevator Group and verifies that all the Escalators that are configured under one group are present under the Group\_Members property of the Elevator Group object.

Configuration Requirements: Configure 2 Escalators (E1 and E2) under the Elevator Group (EG1).

Test Steps:

1. VERIFY (EG1), Group\_Members = (E1, E2)

#### 7.3.2.X45.1.4 Linking of Landing\_Call\_Control Property Test

Purpose: To verify that writing Landing\_Call\_Control property of Elevator Group assigns an active call to the Lift Object linked by pushing it to the Assigned\_Landing\_Calls property of the Lift object.

Test Concept: An Elevator Group is available, and it contains at least one Lift object. Landing\_Call\_Control property of the Elevator Group is written with a Floor number and direction or destination for the lift. Value written to Landing\_Call\_Control property is updated in the Landing\_Calls property of the Elevator Group which in turn updates the Assigned\_Landing\_Calls property of Lift. This test shall be skipped in the event of absence of Landing\_Call\_Control property. If any of the Landing\_Calls or Assigned\_Landing\_Calls property is not present, then the test steps for that specific property shall be skipped.

Configuration Requirements: The Lift (L1) should be present in the Group\_Members property of Elevator Group (EG1). Lowest universal floor number of the lift < A < Highest universal floor number of the lift. Lowest universal floor number of the lift. B = (UP | DOWN | UP\_AND\_DOWN) and C = (B | UP\_AND\_DOWN).

Test Steps:

1. WRITE (EG1), Landing\_Call\_Control = (Floor Number A, Direction B | Destination X)

2. VERIFY (EG1), Landing\_Call\_Control = (Floor Number A, Direction B | Destination X)

3. VERIFY (EG1), Landing\_Calls = (Floor Number A, Direction C | Destination X)

4. VERIFY (L1), Assigned\_Landing\_Calls = (Floor Number A, Direction C)

Notes to Tester: Landing\_Calls property may contain other entries from same lift or different lifts connected under same Elevator Group. If the Elevator Group contains more than 1 lift, value written to Landing\_Call\_Control may get assigned to any other lift, based on the lift algorithm.

## **BTL-TP15.0-0.3.0** Tests for the Escalator object

A device including an Escalator object must claim Protocol\_Revision 18 or higher and must comply with the following section.

[In BTL Checklist, add new Escalator section in existing 3. Object testing.]

Support	Listing	Option
Esca	alator Obje	ct
	R	Base Requirements
	S	Supports writable Out_Of_Service properties
	S	Supports Escalator_Mode property
	0	Supports Energy_Meter_Ref property
	0	Supports CHANGE_OF_STATE event algorithm with Passenger_Alarm property
	0	Supports Reliability_Evaluation_Inhibit property

[In BTL Test Plan, add new Escalator section at end of existing 3. Object testing, with Base Requirements, and five other 3.X46.2 through 3.X46.6 sections as indicated.]

### 3.X46 Escalator Object

### **3.X46.1 Base Requirements**

Base requirements must be met by any IUT that can contain Escalator objects.

BTL - 7.3.2.X46.1.1 Elevator_Group property of Escalator Object linking with Group_Members		
property of Elevator Group Object		
	Test Method	Manual
Configuration As per <i>BTL Specified Tests</i> .		As per BTL Specified Tests.
Test Conditionality Must be executed.		Must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### 3.X46.2 Supports writable Out\_Of\_Service properties

The Out\_Of\_Service property in Escalator objects contained in the IUT is either writable or can be modified by any other means.

BTL - 7.3.2.X43.3 - Out_Of_Service, Status_Flags, and Reliability test for an Object that does not			
conta	contain Present_Value		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	If this property is writable, this test must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 7.3.2.X46.1.2 - Energy_	_Meter, Power_Mode and Operation_Direction Tracking Test	
	Test Method		

	Configuration	As per BTL Specified Tests.
	Test Conditionality	This test must be executed if Energy_Meter or Power_Mode properties
		are present.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X46.1.3 - Passeng	er_Alarm and Fault_Signals Tracking Test
	Test Method	
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X46.1.4 - Escalate	or_Mode Tracking Test
	Test Method	
	Configuration	As per BTL Specified Tests.
	Test Conditionality	This test must be executed if Escalator_Mode property is present.
	Test Directives	
	Testing Hints	
	Notes & Results	

### 3.X46.3 Supports Escalator\_Mode Property

The Escalator\_Mode property in at least one Escalator object is present.

BTL	BTL - 7.3.2.X46.1.5 - Operation_Direction Tracks Escalator_Mode Test	
	Test Method Manual	
Configuration As per <i>BTL Specified Tests</i> .		As per BTL Specified Tests.
	Test ConditionalityMust be executed.	
	Test Directives	
	Testing Hints	
	Notes & Results	

### 3.X46.4 Supports Energy\_Meter\_Ref Property

The Energy\_Meter\_Ref property in at least one Escalator object is present.

BTL	BTL - 7.3.2.X46.1.6 - Energy_Meter_Ref Property Test	
	Test Method Manual	
Configuration As per <i>BTL Specified Tests</i> .		As per BTL Specified Tests.
	Test Conditionality	This test must be executed if both Energy_Meter_Ref and Energy_Meter properties are present.
Test Directives		
	Testing Hints	
	Notes & Results	

### 3.X46.5 Supports CHANGE\_OF\_STATE event algorithm with Passenger\_Alarm property

Intrinsic event algorithm is supported using Passenger\_Alarm property in at least one Escalator.

BTL	TL - 7.3.2.X46.1.7 - CHANGE_OF_STATE for Passenger_Alarm (ConfirmedEventNotification)		
	Test Method	Manual	
	Configuration As per <i>BTL Specified Tests</i> .		
	<b>Test Conditionality</b> This test must be executed if the object under test supports		
CHANGE_OF_STATE event algorithm with Passenger_Alarm		CHANGE_OF_STATE event algorithm with Passenger_Alarm property	
	writable or can be modified by any other means.		
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	BTL - 7.3.2.X46.1.8 - CHANGE_OF_STATE for Passenger_Alarm		
(UnconfirmedEventNotification)		on)	
	Test Method Manual		
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	This test must be executed if the object under test supports	
CHANGE_OF_STATE event algorithm with Passenger_Alarm writable or can be modified by any other means.		CHANGE_OF_STATE event algorithm with Passenger_Alarm property	
		writable or can be modified by any other means.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X46.6 Supports Reliability\_Evaluation\_Inhibit Property

The IUT contains, or can be made to contain, a Reliability\_Evaluation\_Inhibit property that is configurable to a value of TRUE.

BTL	BTL - 7.3.1.X8.1 - Reliability_Evaluation_Inhibit Test	
	Test Method Manual	
	Configuration	As per BTL Specified Tests.
<b>Test Conditionality</b> If no object exists in the IUT for which fault conditions cather then this test shall be skipped.		If no object exists in the IUT for which fault conditions can be generated, then this test shall be skipped.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL - 7.3.1.X8.2 - Reliability_Evaluation_Inhibit Summarization Test		_Evaluation_Inhibit Summarization Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If no object exists in the IUT for which fault conditions can be generated, then this test shall be skipped.
	Test Directives	
	Testing Hints	
	Notes & Results	

[In BTL Specified Tests, add eight new tests 7.3.2.X46.1.1 through 7.3.2.X46.1.8 as indicated.]

# 7.3.2.X46.1.1 Elevator\_Group property of Escalator Object linking with Group\_Members property of Elevator Group Object

Purpose: This test verifies that Elevator\_Group property of Escalator object shall have reference of Elevator Group object whose Group\_Members property contains a reference of Escalator object.

Test Concept: Escalator object falls under one specific Elevator Group object. The reference of Elevator Group object should be mentioned in Elevator\_Group property of Escalator object. If there is no such Elevator Group object, Elevator\_Group property shall contain an object instance of 4194303.

Configuration Requirements: The Escalator (E1), should be present under Elevator Group (EG1). OBJECT is any valid object type.

#### Test Steps:

- 1. VERIFY (E1), Elevator\_Group = (EG1)
- 2. VERIFY (EG1), Group Members =  $((E1), \ldots, En)$
- 3. IF (IUT does not contain reference of any Elevator Group Object) THEN VERIFY (E1), Elevator\_Group = (OBJECT, 4194303)

#### 7.3.2.X46.1.2 Energy\_Meter, Power\_Mode and Operation\_Direction Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Escalator object, it does not track the changes made for Energy\_Meter, Power\_Mode and Operation\_Direction property and it does not control the escalator operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Energy\_Meter, Power\_Mode and Operation\_Direction property shall not make escalator to update its energy value, power mode and operation direction. Also, while making escalator's energy, power mode and operation direction change from current status, it shall not get updated to Energy\_Meter, Power\_Mode and Operation\_Direction property of the Escalator object. Out\_Of\_Service property of the Escalator object is set to TRUE in the beginning of the test. If either of the Energy\_Meter or Power\_Mode properties are not present, then the test steps for that specific property shall be skipped.

Configuration Requirements: The Escalator Object supports Energy\_Meter and/or Power\_Mode properties. Escalator Power\_Mode is TRUE and Operation\_Direction is STOPPED. Escalator is having energy meter value = X. Tester shall select any value for energy meter Y; Y < 99999 or permitted by IUT. Tester shall select any Operation\_Direction supported by IUT while testing.

- IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE
  - MAKE (Out Of Service = TRUE)
- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Energy Meter = Y
- 5. VERIFY Energy\_Meter = Y
- 6. CHECK (the escalator's energy consumption is having value = X or value other than Y)
- 7. MAKE (the escalator's energy consumption value = Z)
- 8. VERIFY Energy\_Meter = Y
- 9. WRITE Power\_Mode = FALSE
- 10. VERIFY Power\_Mode = FALSE
- 11. CHECK (the escalator is still powered up independent of the value written)
- 12. MAKE (the escalator's power mode to be TRUE from FALSE)
- 13. VERIFY Power\_Mode = FALSE
- 14. WRITE Operation\_Direction = UP\_RATED\_SPEED
- 15. VERIFY Operation\_Direction = UP\_RATED\_SPEED

- 16. CHECK (the escalator remains stopped)
- 17. MAKE (the escalator's operation direction to be DOWN\_RATED\_SPEED)
- 18. VERIFY Operation\_Direction = UP\_RATED\_SPEED
- 19. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = FALSE ELSE
- MAKE (Out\_Of\_Service = FALSE)

20. VERIFY Out\_Of\_Service = FALSE

21. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X46.1.3 Passenger\_Alarm and Fault\_Signals Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Escalator object, it does not track the changes made for Passenger\_Alarm and Fault\_Signals property and it does not control the escalator operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Passenger\_Alarm and Fault\_Signals property shall not make escalator to update its alarm and fault status. Also, while making escalator's fault and alarm status change from current value, it shall not get updated to Passenger\_Alarm and Fault\_Signals property of the Escalator object. Out\_Of\_Service property of the Escalator object is set to TRUE in the beginning of the test. If Fault\_Signals property is not present, then the respective test steps shall be skipped.

Configuration Requirements: Escalator has no alarm or fault at the start of test. Tester shall select any value for Fault\_Signals property testing that is supported by IUT.

Test Steps:

1. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE

ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Passenger\_Alarm = TRUE
- 5. VERIFY Passenger\_Alarm = TRUE
- 6. CHECK (the escalator's alarm is not triggered)
- 7. MAKE (the escalator in NORMAL state)
- 8. VERIFY Passenger\_Alarm = TRUE
- 9. WRITE Fault\_Signals = OVERSPEED\_FAULT
- 10. VERIFY Fault\_Signals = OVERSPEED\_FAULT
- 11. CHECK (the escalator does not have any fault into it)
- 12. MAKE (the escalator to have SAFETY\_DEVICE\_FAULT fault)
- 13. VERIFY Fault\_Signals = OVERSPEED\_FAULT
- 14. IF (Out\_Of\_Service is writable) THEN

WRITE Out\_Of\_Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X46.1.4 Escalator\_Mode Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Escalator object, it does not track the changes made for Escalator\_Mode property and also it does not control the escalator operation from this property.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Escalator\_Mode property shall not make escalator to update its mode. Also, while making escalator's mode to change from current value, it shall not get updated to Escalator\_Mode property of the Escalator object. Out\_Of\_Service property of the Escalator object is set to TRUE in the beginning of the test. If this property is not present, then this test shall be skipped.

Configuration Requirements: The Escalator Object shall support Escalator\_Mode property. Escalator runs at UP mode. Tester shall select any value for Escalator\_Mode property for testing that are supported by IUT.

Test Steps:

- 1. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = TRUE

ELSE

- MAKE (Out\_Of\_Service = TRUE)
- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Escalator\_Mode = DOWN
- 5. VERIFY Escalator\_Mode = DOWN
- 6. CHECK (the escalator or slanted passenger conveyor is still moving upward)
- 7. MAKE (the escalator to move from downward to upward)
- 8. VERIFY Escalator\_Mode = DOWN
- 9. IF (Out\_Of\_Service is writable) THEN WRITE Out Of Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 10. VERIFY Out\_Of\_Service = FALSE
- 11. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X46.1.5 Operation\_Direction Tracks Escalator\_Mode Test

Purpose: To verify the linking of Operation\_Direction property and Escalator\_Mode property of Escalator object

Test Concept: Operation\_Direction property i.e. the direction and speed in which this escalator is presently moving corresponds to the Escalator\_Mode property of Escalator object

Test Steps:

- 1. IF (Escalator\_Mode = STOP) THEN VERIFY Operation\_Direction = STOPPED
- 2. IF (Escalator\_Mode = UP) THEN VERIFY Operation\_Direction = UP\_RATED\_SPEED | UP\_REDUCED\_SPEED
- 3. IF (Escalator\_Mode = DOWN) THEN VERIFY Operation\_Direction = DOWN\_RATED\_SPEED | DOWN\_REDUCED\_SPEED

#### 7.3.2.X46.1.6 Energy\_Meter\_Ref Property Test

Purpose: To verify linking of Energy\_Meter property and Energy\_Meter\_Ref property.

Test Concept: If the Energy\_Meter\_Ref property is present and initialized with and Object (contains an instance other than 4194303), then the Energy\_Meter property, if present, shall have a value of 0.0. If Energy\_Meter\_Ref property is un-initialized, then the Energy\_Meter property shall have any valid value.

Test Steps:

1. IF (Energy\_Meter\_Ref is present and initialized with instance other than 4194303) THEN

VERIFY Energy\_Meter = 0.0 ELSE VERIFY Energy\_Meter = (Any Valid Value)

### 7.3.2.X46.1.7 CHANGE\_OF\_STATE for Passenger\_Alarm (ConfirmedEventNotification)

Purpose: To verify the correct operation of the CHANGE\_OF\_STATE event algorithm. This test applies to Event Enrollment objects with an Event\_Type of CHANGE\_OF\_STATE and to intrinsic event reporting for Escalator and Lift objects.

Test Concept: The object begins the test in a NORMAL state. pMonitoredValue is set to TRUE. After pTimeDelay the object shall enter the OFFNORMAL state and transmit an event notification message. pMonitoredValue is set to FALSE corresponding to a NORMAL state. After pTimeDelayNormal the object shall enter the NORMAL state and transmit an event notification message

Configuration Requirements: The IUT shall be configured such that the Event\_Enable property has a value of TRUE for the TO-OFFNORMAL, TO-FAULT and TO-NORMAL transitions. The Issue\_Confirmed\_Notifications parameter shall have a value of TRUE. The event-generating objects shall be in a NORMAL state at the start of the test. If a Notification Class object is being used to configure recipient information the value of the Transitions parameter for all recipients shall be (TRUE, TRUE, TRUE). If present in the object being tested, the Event\_Detection\_Enable property shall have a value of TRUE, Event\_Algorithm\_Inhibit shall have a value of FALSE.

1. VERIFY pCurrentState = NORMAL			
2. I F (the object, or referenced object, if using Event Enrollment, is an Escalator or Lift object with			
Passenger_Alarm property) THEN			
3. MAKE (pMonitoredValue (Passenger_Alarm) = 7	. MAKE (pMonitoredValue (Passenger_Alarm) = TRUE)		
4. WAIT (pTimeDelay)			
5. BEFORE Notification Fail Time			
<b>RECEIVE</b> ConfirmedEventNotification-Rec	quest,		
'Process Identifier' =	(any valid process ID),		
'Initiating Device Identifier' =	IUT,		
'Event Object Identifier' = (the intr	insic reporting object being tested or the EventEnrollment		
object being tested),			
'Time Stamp' =	(T1, the current local time or sequence number),		
'Notification Class' =	(the configured notification class),		
'Priority' = (the value config	ured to correspond to a TO-OFFNORMAL transition),		
'Event Type' =	CHANGE_OF_STATE,		
'Message Text' =	(optional, any valid message text),		
'Notify Type' =	EVENT   ALARM,		
'AckRequired' =	TRUE   FALSE,		
'From State' =	NORMAL,		
'To State' =	OFFNORMAL,		
'Event Values' =	(pMonitoredValue, pStatusFlags)		
6. TRANSMIT BACnet-SimpleACK-PDU			
7. VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)			
8. VERIFY pCurrentState = OFFNORMAL			
9. VERIFY Event_Time_Stamps = (T1, *, *)			
10. MAKE (pMonitoredValue (Passenger_Alarm) = FALSE)			
11. WAIT (pTimeDelayNormal)			
12. BEFORE Notification Fail Time			
RECEIVE ConfirmedEventNotification-Rec	quest,		
'Process Identifier' =	(any valid process ID),		
'Initiating Device Identifier' =	IUT		

'Event Object Identifier' =	(the intrinsic reporting object being tested or the
	EventEnrollment object being tested),
'Time Stamp' =	(T2, the current local time or sequence number),
'Notification Class' =	(the configured notification class),
'Priority' =	(the value configured to correspond to a TO-NORMAL
	transition),
'Event Type' =	CHANGE_OF_STATE,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	EVENT   ALARM,
'AckRequired' =	TRUE   FALSE,
'From State' =	OFFNORMAL,
'To State' =	NORMAL,
'Event Values' =	(pMonitoredValue, pStatusFlags)
13. TRANSMIT BACnet-SimpleACK-PDU	
14. VERIFY pStatusFlags = (FALSE, FALSE, ?, ?)	
15. VERIFY pCurrentState = NORMAL	
16. VERIFY Event_Time_Stamps = (T1, *, T2)	

#### 7.3.2.X46.1.8 CHANGE\_OF\_STATE for Passenger\_Alarm (UnconfirmedEventNotification)

Purpose: To verify the correct operation of the CHANGE\_OF\_STATE event algorithm. This test applies to Event Enrollment objects with an Event\_Type of CHANGE\_OF\_STATE and to intrinsic event reporting for Escalator and Lift objects.

Test Concept: The object begins the test in a NORMAL state. pMonitoredValue is set to TRUE. After pTimeDelay the object shall enter the OFFNORMAL state and transmit an event notification message. pMonitoredValue is set to FALSE corresponding to a NORMAL state. After pTimeDelayNormal the object shall enter the NORMAL state and transmit an event notification message

Configuration Requirements: The IUT shall be configured such that the Event\_Enable property has a value of TRUE for the TO-OFFNORMAL, TO-FAULT and TO-NORMAL transitions. The Issue\_Confirmed\_Notifications parameter shall have a value of FALSE. The event-generating objects shall be in a NORMAL state at the start of the test. If a Notification Class object is being used to configure recipient information the value of the Transitions parameter for all recipients shall be (TRUE, TRUE, TRUE). If present in the object being tested, the Event\_Detection\_Enable property shall have a value of TRUE, Event\_Algorithm\_Inhibit shall have a value of FALSE.

Test Steps: The test steps for this test are identical to the test steps in 7.3.2.X46.1.7 except that the ConfirmedEventNotification requests are UnconfirmedEventNotification requests and the TD does not acknowledge receiving the notifications.

## **BTL-TP15.0-0.4.0** Tests for the Lift object

A device including a Lift object must claim Protocol\_Revision 18 or higher and must comply with the following section.

Support	Listing	Option
Lift	Object	
	R	Base Requirements
	S	Supports writable Out_Of_Service properties
	S	Supports Landing_Door_Status and Car_Door_Status properties
	Ο	Supports Making_Car_Call, and Register_Car_Call properties
	Ο	Supports BACnetARRAY Properties related to the doors of a car
	Ο	Supports Car_Position and Next_Stopping_Floor properties
	Ο	Supports Assigned_Landing_Calls, Making_Car_Call and Registered_Car_Call properties
	Ο	Supports Energy_Meter_Ref and Energy_Meter properties
	Ο	Supports Higher_Deck and Lower_Deck properties
	Ο	Supports Reliability_Evaluation_Inhibit property
	0	Supports Reliability Evaluation
	0	Supports CHANGE_OF_STATE event algorithm with Passenger_Alarm property
	0	Supports writable Assigned_Landing_Calls property

[In BTL Checklist, add new Lift section in existing 3]

[In BTL Test Plan, add new Lift section at end of existing 3. Object testing, with sections 3.X47.1 Base Requirements, and twelve other 3.X47.2 through 3.X47.13 sections as indicated.

### 3.X47 Lift Object

### **3.X47.1 Base Requirements**

Base requirements must be met by any IUT that can contain Lift objects.

BTL	BTL - 7.3.2.X47.1.1 - Elevator_Group property of Lift Object linking with Group_Members		
prope	erty of Elevator Group (	Dbject.	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X47.2 Supports writable Out\_Of\_Service properties

The Out\_Of\_Service property in Lift objects contained in the IUT is either writable or can be modified by any other means.

BTL	BTL - 7.3.2.X43.3 - Out_Of_Service, Status_Flags, and Reliability test for an Object that does not		
conta	contain Present_Value		
	Test Method	Manual	

1	Configuration	This test shall be executed using a Lift object
	Test Conditionality	If this property is writable, this test must be executed
	Tost Directives	IT this property is writable, this lest must be excedded.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.2 - Car_M	oving_Direction and Car_Assigned_Direction Tracking Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.3 - Car Do	or Status and Landing Door Status Tracking Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b>
	Test Conditionality	If Out Of Service property is either writable or can be modified by other
	Test Conditionality	means and if any of these properties are present this test must be
		executed
	Tast Directives	
	Test Directives	
	Notos & Dosults	
DTI	<b>Notes &amp; Results</b>	
BIL	- 7.3.2.X47.1.4 - Car_Po	sition and Next_Stopping_Floor Tracking Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.5 - Passeng	er_Alarm and Fault_Signals Tracking Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.6 - Making	_Car_Call, Car_Mode & Car_Door_Command Tracking Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If Out Of Service property is either writable or can be modified by other
	· ·	means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.7 - Assigne	d Landing Call and Registered Car Call Tracking Test
	Test Method	Manual
	Configuration	As per RTI Specified Tests
1	Conngulation	

	Test Conditionality	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.8 - Car_Do	or_Zone and Car_Load Tracking Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.2.X47.1.9 - Energy_	_Meter and Car_Drive_Status Tracking Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If Out_Of_Service property is either writable or can be modified by other
		means and if any of these properties are present, this test must be
		executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

### 3.X47.3 Supports Making\_Car\_Call and Register\_Car\_Call Properties

Either of the Making\_Car\_Call, Register\_Car\_Call properties in at least one Lift object are present.

BTL	BTL - 7.3.2.X47.1.10 - Making_Car_Call and Registered_Car_Call Tests		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if Making_Car_Call and Registered_Car_Call properties are present.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X47.4 Supports BACnetARRAY Properties related to the doors of a car

BACnetARRAY properties related to the doors of a car are present in at least one Lift object.

BTL	BTL - 7.3.2.X47.1.11 - Array Size of the Lift Object properties based on car door size		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if any of the BACnetARRAY properties	
	-	Car_Door_Text, Assigned_Landing_Calls, Making_Car_Call,	
		Registered_Car_Call, Car_Door_Status, Car_Door_Command and	
		Landing_Door_Status are present.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X47.5 Supports Landing Door\_Status and Car\_Door\_Status Properties

BTL	BTL - 7.3.2.X47.1.12 - Landing_Door_Status Tracks Car_Door_Status Test	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	This test must be executed if Landing Door Status property is present.
	Test Directives	
	Testing Hints	
	Notes & Results	

The Landing\_Door\_Status property in at least one Lift object is present.

### 3.X47.6 Supports Car\_Position and Next\_Stopping\_Floor Properties

BTL - 7.3.2.X47.1.13 - Highest Universal floor number linking to Car Position and Next\_Stopping\_Floor properties **Test Method** Manual Configuration As per BTL Specified Tests. This test must be executed if Car Position and Next Stopping Floor **Test Conditionality** properties are present. If any property is not present, the respective step shall be skipped **Test Directives Testing Hints** Notes & Results

# Either of the Car\_Position,Next\_Stopping\_Floor property in at least one Lift object is present.

### 3.X47.7 Supports Assigned\_Landing\_Calls, Making\_Car\_Call and **Registered Car Call Properties**

Either of the Assigned\_Landing\_Calls, Making\_Car\_Call and Register\_Car\_Call property in at least one Lift object is present.

BTL	BTL - 7.3.2.X47.1.14 Highest Universal floor number linking to Assigned_Landing_Calls,		
Maki	ng_Car_Call and Regist	tered_Car_Call properties	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	This test must be executed if Assigned_Landing_Calls,	
		Making_Car_Call and Registered_Car_Call properties are present. If any property is not present, the respective step shall be skipped	
	Test Directives		
	Testing Hints		
Notes & Results			

### **3.X47.8** Supports Energy Meter Ref and Energy Meter Properties

The Energy\_Meter\_Ref and Energy\_Meter property in at least one Lift object is present.

BTL ·	TL - 7.3.2.X47.1.15 Energy_Meter_Ref Property Tests		
	Test Method Manual		
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if Energy_Meter_Ref and Energy_Meter	
	-	property is present	

	Test Directives	
	Testing Hints	
	Notes & Results	

### **3.X47.9** Supports Higher\_Deck and Lower\_Deck Properties

The Higher\_Deck and Lower\_Deck properties in at least one Lift object is present.

BTL ·	BTL - 7.3.2.X47.1.16 Higher_Deck and Lower_Deck Tests		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if Higher_Deck and Lower_Deck properties	
		are present	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X47.10 Supports Reliability\_Evaluation\_Inhibit Property

The IUT contains, or can be made to contain, a Reliability\_Evaluation\_Inhibit property that is configurable to a value of TRUE.

BTL - 7.3.1.X8.1 - Reliability_Evaluation_Inhibit Test		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If no object exists in the IUT for which fault conditions can be generated,
		then this test shall be skipped.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL - 7.3.1.X8.2 - Reliability_Evaluation_Inhibit Summarization Test		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If no object exists in the IUT for which fault conditions can be
		generated, then this test shall be skipped.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### **3.X47.11 Supports Reliability Evaluation**

The IUT contains, or can be made to contain, a Lift object that can generate ConfirmedEventNotifications and UnconfirmedEventNotifications with an Event\_Type of CHANGE\_OF\_RELIABILITY.

BTL (Conf	BTL - 8.4.X1.13 Change_Of_Reliability with FAULT_LISTED Algorithm (ConfirmedEventNotification)	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	This test must be executed
	Test Directives	
	Testing Hints	

Notes & Results			
BTL - 8.4.X1.14 Change_Of_Reliability with FAULT_LISTED Algorithm			
(UnconfirmedEventNotification)			
Test Method	Manual		
Configuration	As per BTL Specified Tests.		
Test Conditionality	This test must be executed		
Test Directives			
Testing Hints			
Notes & Results			

### 3.X47.12 Supports CHANGE\_OF\_STATE event algorithm with Passenger\_Alarm property

Intrinsic event algorithm is supported using Passenger\_Alarm property in at least one Lift object.

BTL	BTL - 7.3.2.X46.1.8 CHANGE_OF_STATE for Passenger_Alarm (ConfirmedEventNotification)		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if the object under test supports	
		CHANGE_OF_STATE event algorithm with Passenger_Alarm property	
		writable or can be modified by any other means.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	BTL - 7.3.2.X46.1.9 CHANGE_OF_STATE for Passenger_Alarm (UnconfirmedEventNotification)		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test must be executed if the object under test supports	
		CHANGE_OF_STATE event algorithm with Passenger_Alarm property	
		writable or can be modified by any other means.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X47.13 Supports writable Assigned\_Landing\_Calls Property

The Assigned\_Landing\_Calls property is present in at least one Lift object.

BTL - 7.3.2.X47.1.17 - Linking of Assigned_Landing_Calls property of Lift Object to				
Landing_Calls property of Elevator Group				
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	This test must be executed if Assigned Landing Calls is writable.		
	<b>Test Directives</b>			
	Testing Hints			
	Notes & Results			

[In BTL Specified Tests, add the following new tests]

7.3.2.X47.1.1 Elevator\_Group property of Lift Object linking with Group\_Members property of Elevator Group Object.

Purpose: This test verifies that Elevator\_Group property of Lift object shall have reference of Elevator Group object whose Group\_Members property contains a reference of Lift object.

Test Concept: Lift object falls under one specific Elevator Group object. The reference of Elevator Group object should be mentioned in Elevator\_Group property of Lift object. If there is no such Elevator Group object, Elevator\_Group property shall contain an object instance of 4194303.

Configuration Requirements: The Lift (L1) should present under the Elevator Group (EG1). OBJECT is any valid object type.

#### Test Steps:

- 1. VERIFY (L1), Elevator\_Group = (EG1)
- 2. VERIFY (EG1), Group\_Members = ((L1), ..., Ln)
- 3. IF (IUT does not have reference of any such Elevator Group object) THEN VERIFY (L1), Elevator\_Group = (OBJECT, 4194303)

#### 7.3.2.X47.1.2 Car\_Moving\_Direction and Car\_Assigned\_Direction Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Car\_Moving\_Direction and Car\_Assigned\_Direction property and it does not control the lift operation from these properties.

Test Concept: When Out\_Of\_Service is set to TRUE, writing Car\_Moving\_Direction and Car\_Assigned\_Direction property shall not make lift to serve specified direction. Also, making lift to serve any direction shall not be updated in Car\_Moving\_Direction and Car\_Assigned\_Direction property of Lift object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If Car\_Assigned\_Direction property is not present, then the respective test steps shall be skipped.

Configuration Requirements: 'X' and 'Y' are any valid directions supported by IUT. Tester shall select any car moving direction and car assigned direction supported by IUT.

Test Steps:

- 1. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = TRUE

ELSE

- MAKE (Out\_Of\_Service = TRUE)
- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Car\_Moving\_Direction = Direction X
- 5. VERIFY Car\_Moving\_Direction = Direction X
- 6. CHECK (the lift is not serving as per the Car\_Moving\_Direction property)
- 7. MAKE (the lift to move in Direction Y)
- 8. VERIFY Car\_Moving\_Direction = Direction X
- 9. WRITE Car\_Assigned\_Direction = Direction X
- 10. VERIFY Car\_Assigned\_Direction = Direction X
- 11. CHECK (the lift is not serving as per the Car\_Assigned\_Direction property)
- 12. MAKE (the lift assigned towards Direction Y)
- 13. VERIFY Car\_Assigned\_Direction = Direction X
- 14. IF (Out\_Of\_Service is writable) THEN

WRITE Out\_Of\_Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.3 Car\_Door\_Status and Landing\_Door\_Status Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Car\_Door\_Status and Landing\_Door\_Status property and it does not control the lift operation from these properties.

Test Concept: When Out\_Of\_Service is set to TRUE, writing Car\_Door\_Status and Landing\_Door\_Status property shall not make lift and landing doors to operate. Also, making lift and landing doors to operate shall not be updated in Car\_Door\_Status and Landing\_Door\_Status property when the Out\_Of\_Service is set to TRUE. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If Landing\_Door\_Status property is not present, then the respective test steps shall be skipped.

Configuration Requirements: Lift's Door starts in OPEN State. ARRAY INDEX = (any valid value N;  $1 \le N \le$  number of doors of a car). Universal floor number = (X = any valid floor number of the lift connected to the IUT) Tester shall select any car door status and landing door status values supported by IUT.

Test Steps:

1. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Car\_Door\_Status = CLOSED, ARRAY INDEX = N
- 5. VERIFY Car\_Door\_Status = CLOSED, ARRAY INDEX = N
- 6. CHECK (the lift's car door is not operating as per the Car\_Door\_Status property)
- 7. MAKE (the lift's car door N to OPEN)
- 8. VERIFY Car\_Door\_Status = CLOSED, ARRAY INDEX = N
- 9. WRITE Landing\_Door\_Status = CLOSING, ARRAY INDEX = N, Universal floor number = X
- 10. VERIFY Landing\_Door\_Status = CLOSING, ARRAY INDEX = N
- 11. CHECK (the specified landing door is not serving as per the Landing\_Door\_Status property)
- 12. MAKE (the landing door for car door N to OPEN at Universal floor number X)
- 13. VERIFY Landing\_Door\_Status = CLOSING, ARRAY INDEX = N, Universal floor number = X
- 14. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.4 Car\_Position and Next\_Stopping\_Floor Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made in Car\_Position and Next\_Stopping\_Floor property and also it does not control the lift operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Car\_Position and Next\_Stopping\_Floor property shall not make lift to update its car position and next stopping floor. Also, while making lift's car position and next stopping floor change from current value, it shall not get updated to Car\_Position and Next\_Stopping\_Floor property of the Lift object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If Next\_Stopping\_Floor property is not present, then the respective test steps shall be skipped.

Configuration Requirements: Lift's current position (floor) is A. Universal floor number = (X, Y, A, B, C = any valid floor number of the lift connected to the IUT). Tester shall select any floor number supported by IUT for this test.

Test Steps:

1. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Car\_Position = Y
- 5. VERIFY Car\_Position = Y
- 6. CHECK (the lift still stands at the floor A)
- 7. MAKE (the lift to stand at the floor X)
- 8. VERIFY Car\_Position = Y
- 9. WRITE Next\_Stopping\_Floor = C
- 10. VERIFY Next\_Stopping\_Floor = C
- 11. CHECK (the lift is not moving towards floor C and it still stands at floor X)
- 12. MAKE (the lift to move from floor X to reach floor B)
- 13. VERIFY Next\_Stopping\_Floor = C
- 14. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.5 Passenger\_Alarm and Fault\_Signals Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Passenger\_Alarm and Fault\_Signals property and it does not control the lift operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Passenger\_Alarm and Fault\_Signals property shall not make lift to update its alarm and fault status. Also, while making lift's fault and alarm status change from current value, it shall not get updated to Passenger\_Alarm and Fault\_Signals property of the Lift object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If Fault\_Signals property is not present, then the respective test steps shall be skipped.

Configuration Requirements: Lift has no alarm or fault at the start of test. Tester shall select any value for Fault\_Signals property testing that is supported by IUT.

Test Steps:

1. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. WRITE Passenger\_Alarm = TRUE
- 4. VERIFY Passenger\_Alarm = TRUE
- 5. CHECK (the lift's alarm is not triggered)
- 6. MAKE (the lift to move from Alarm to normal state)
- 7. VERIFY Passenger\_Alarm = TRUE

- 8. WRITE Fault\_Signals = CALL\_BUTTON\_STUCK
- 9. VERIFY Fault\_Signals = CALL\_BUTTON\_STUCK
- 10. CHECK (the lift does not have any fault into it)
- 11. MAKE (the lift to have POSITION\_LOST fault)
- 12. VERIFY Fault\_Signals = CALL\_BUTTON\_STUCK
- 13. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = FALSE ELSE

MAKE (Out\_Of\_Service = FALSE)

14. VERIFY Out\_Of\_Service = FALSE

### 7.3.2.X47.1.6 Making\_Car\_Call, Car\_Mode & Car\_Door\_Command Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Making\_Car\_Call, Car\_Mode & Car\_Door\_Command property and also it does not control the lift operation from these properties.

Test Concept: When Out\_Of\_Service is set to TRUE, writing Making\_Car\_Call, Car\_Mode & Car\_Door\_Command property shall not make lift to serve specified floor, to set the mode and to execute car door commands. Also, making lift to serve different floors, to operate at different modes and for various car door commands shall not be updated in Making\_Car\_Call, Car\_Mode & Car\_Door\_Command properties of Lift Object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If any of the Making\_Car\_Call, Car\_Mode or Car\_Door\_Command property is not present, then the test steps for that specific property shall be skipped.

Configuration Requirements: Car\_Mode is NORMAL and Car\_Door\_Command is CLOSE at the start of the test. ARRAY INDEX = (any valid value N;  $1 \le N \le$  number of doors of a car). Universal floor number = (X, Y = any valid floor number of the lift connected to the IUT). Tester shall select any car door command or car mode supported by IUT while testing.

Test Steps:

- 1. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE
  - ELSE
    - MAKE (Out\_Of\_Service = TRUE)
- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Making\_Car\_Call = any valid floor X, ARRAY INDEX = N
- 5. VERIFY Making\_Car\_Call = X, ARRAY INDEX = N
- 6. CHECK (the lift is not serving as per value X in Making\_Car\_Call property)
- 7. MAKE (the lift to serve call at floor Y for car door N)
- 8. VERIFY Making\_Car\_Call = X, ARRAY INDEX = N
- 9. WRITE Car\_Door\_Command = OPEN, ARRAY INDEX = N
- 10. VERIFY Car\_Door\_Command = OPEN, ARRAY INDEX = N
- 11. CHECK (the lift's car door N is not opening as per the Car\_Door\_Command property)
- 12. MAKE (the lift to CLOSE at the car door N from OPEN or NONE)
- 13. VERIFY Car\_Door\_Command = OPEN, ARRAY INDEX = N
- 14. WRITE Car\_Mode = HOMING
- 15. VERIFY Car\_Mode = HOMING
- 16. CHECK (the lift is not moving into HOMING mode)
- 17. MAKE (the lift into PARKING mode)
- 18. VERIFY Car\_Mode = HOMING
- 19. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

20. VERIFY Out\_Of\_Service = FALSE

21. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.7 Assigned\_Landing\_Call and Registered\_Car\_Call Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Assigned\_Landing\_Call and Registered\_Car\_Call property and it does not control the lift operation from these properties.

Test Concept: When Out\_Of\_Service is set to TRUE, writing Assigned\_Landing\_Call and Registered\_Car\_Call property shall not make lift to serve specified floors and direction. Also, making lift to serve any floors and direction shall not be updated in Assigned\_Landing\_Calls and Registered\_Car\_Call property of Lift object. . Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If any of the Assigned\_Landing\_Calls and Registered\_Car\_Call property shall be skipped.

Configuration Requirements: ARRAY INDEX = (any valid value N;  $1 \le N \le$  number of doors of a car). Universal floor number = (A, B, X1...n, Y1...n = any valid floor number of the lift connected to the IUT). P, Q is any valid direction supported by IUT.

Test Steps:

- 1. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = TRUE

ELSE

- MAKE (Out\_Of\_Service = TRUE)
- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Assigned\_Landing\_Calls = (Floor A, Direction P), ARRAY INDEX = N
- 5. VERIFY Assigned\_Landing\_Calls = (Floor A, Direction P), ARRAY INDEX = N
- 6. CHECK (the lift is not serving as per the values of Assigned\_Landing\_Calls property)
- 7. MAKE (the lift to serve landing call at Floor B, Direction Q for car door N)
- 8. VERIFY Assigned\_Landing\_Calls = (Floor A, Direction P), ARRAY INDEX = N
- 9. WRITE Registered\_Car\_Call = (X1, X2, X3, X4...Xn), ARRAY INDEX = N
- 10. VERIFY Registered\_Car\_Call = (X1, X2, X3, X4...Xn), ARRAY INDEX = N
- 11. CHECK (the lift is not serving as per the Registered\_Car\_Call property)
- 12. MAKE (the lift to serve calls at Floor (Y1, Y2, Y3....Yn) for car door N)
- 13. VERIFY Registered\_Car\_Call = (X1, X2, X3, X4...Xn), ARRAY INDEX = N
- 14. IF (Out\_Of\_Service is writable) THEN
  - WRITE Out\_Of\_Service = FALSE
  - ELSE
    - MAKE (Out\_Of\_Service = FALSE)
- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

### 7.3.2.X47.1.8 Car\_Door\_Zone and Car\_Load Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Car\_Door\_Zone and Car\_Load property and it does not control the lift operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Car\_Door\_Zone and Car\_Load property shall not make lift update its car door zone and its load. Also, while making lift's car to enter to a particular door zone where door opening is permitted and having a specific weight of lift car shall not get updated to Car\_Door\_Zone and Car\_Load properties of the Lift object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If any of the Car\_Door\_Zone and Car\_Load property is not present, then the test steps for that specific property shall be skipped.

Configuration Requirements: Lift is stopped at any floor in the specified car door zone and having X units of weight. Tester shall select any weight within the permissible limit of the IUT while testing the Car\_Load property.

Test Steps:

 IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE

MAKE (Out\_Of\_Service = TRUE)

- 2. VERIFY Out\_Of\_Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Car\_Door\_Zone = FALSE
- 5. VERIFY Car\_Door\_Zone = FALSE
- 6. CHECK (the lift's car door zone remains unchanged independent of value written)
- 7. MAKE (the lift's car door to door opening permitted zone)
- 8. VERIFY Car\_Door\_Zone = FALSE
- 9. WRITE Car\_Load = X+1 units
- 10. VERIFY Car\_Load = X+1 units
- 11. CHECK (the car load is X units)
- 12. MAKE (the lift car load to X+2)
- 13. VERIFY Car\_Load = X+1 units
- 14. IF (Out\_Of\_Service is writable) THEN WRITE Out Of Service = FALSE

ELSE

MAKE (Out\_Of\_Service = FALSE)

- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.9 Energy\_Meter and Car\_Drive\_Status Tracking Test

Purpose: To verify that when Out\_Of\_Service property is set to TRUE for the monitored Lift object, it does not track the changes made for Energy\_Meter and Car\_Drive\_Status property and it does not control the lift operation from these properties.

Test Concept: When the Out\_Of\_Service is set to TRUE, writing Energy\_Meter and Car\_Drive\_Status property shall not make lift to update its energy value and car drive status. Also, while making lift's energy and car drive status change from current value, it shall not get updated to Energy\_Meter and Car\_Drive\_Status property of the Lift object. Out\_Of\_Service property of the Lift object is set to TRUE in the beginning of the test. If any of the Energy\_Meter and Car\_Drive\_Status property shall be skipped.

Configuration Requirements: Lift is stopped at any floor, i.e. car drive status is stationary. Lift is having energy meter value = X. Tester shall select any value for energy meter Y; Y < 99999 or permitted by IUT. Tester shall select any car drive status supported by IUT.

- IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = TRUE ELSE MAKE (Out\_Of\_Service = TRUE)
- 2. VERIFY Out Of Service = TRUE
- 3. VERIFY Status\_Flags = (?, ?, ?, TRUE)
- 4. WRITE Energy\_Meter = Y
- 5. VERIFY Energy\_Meter = Y

- 6. CHECK (the lift's energy consumption is having value = X or value other than Y)
- 7. MAKE (the lift's energy consumption value = Z)
- 8. VERIFY Energy\_Meter = Y
- 9. WRITE Car\_Drive\_Status = BRAKING
- 10. VERIFY Car\_Drive\_Status = BRAKING
- 11. CHECK (the lift's car drive status is STATIONARY)
- 12. MAKE (the lift's car drive status to ACCELERATE)
- 13. VERIFY Car\_Drive\_Status = BRAKING
- 14. IF (Out\_Of\_Service is writable) THEN WRITE Out\_Of\_Service = FALSE ELSE
  - MAKE (Out\_Of\_Service = FALSE)
- 15. VERIFY Out\_Of\_Service = FALSE
- 16. VERIFY Status\_Flags = (?, ?, ?, FALSE)

#### 7.3.2.X47.1.10 Making\_Car\_Call and Registered\_Car\_Call Test

Purpose: To verify that the values written into Making\_Car\_Call property of lift object reflects in its Registered\_Car\_Call property at the same door side array index.

Test Concept: Making\_Car\_Call property of Lift (L1) object being tested is subjected for car calls provided by means of passenger requesting for car stop or by means of writing the property. The Registered\_Car\_Call property value at a specified array index is checked to verify that it is same as that of value provided to Making\_Car\_Call property.

Configuration Requirements: For below steps 'Array Index' = (any valid value N;  $1 \le N \le$  number of doors of a car) and 'Property Value' = (any valid value X; X \le highest universal floor number of the lift)

Test Steps:

 IF (Making\_Car\_Call is writable) THEN WRITE (L1), Making\_Car\_Call = X, ARRAY INDEX = N ELSE
 MAKE (Making\_Car\_Call = (Value of X)\_ARRAY INDEX = N

MAKE (Making\_Car\_Call = (Value of X), ARRAY INDEX = N)

- 2. VERIFY (L1), Making\_Car\_Call = X, ARRAY INDEX = N
- 3. VERIFY (L1), Registered\_Car\_Call = X, ARRAY INDEX = N

Notes to Tester: Registered\_Car\_Call property may contain other additional entries.

#### 7.3.2.X47.1.11 Array Size of the Lift Object properties based on car door size.

Purpose: To verify that the size of the Car\_Door\_Text, Assigned\_Landing\_Calls, Making\_Car\_Call, Registered\_Car\_Call, Car\_Door\_Status, Car\_Door\_Command and Landing\_Door\_Status array corresponds to the number of car doors present in the lift car and all are of same size.

Test Concept: Above properties will be verified for the array index 0 equals the number of car doors present in the Lift (L1). If change of car door size is possible, change and REPEAT all the steps else skip. If any of above properties are not present, then skip and proceed with the test for available properties.

- 1. VERIFY (L1), Car\_Door\_Text = (Number of car doors present in the Lift), ARRAY INDEX = 0
- 2. VERIFY (L1), Assigned\_Landing\_Calls = (Number of car doors present in Lift), ARRAY INDEX = 0
- 3. VERIFY (L1), Making\_Car\_Call = (Number of car doors present in the Lift), ARRAY INDEX = 0
- 4. VERIFY (L1), Registered\_Car\_Call = (Number of car doors present in the Lift), ARRAY INDEX = 0
- 5. VERIFY (L1), Car\_Door\_Status = (Number of car doors present in the Lift), ARRAY INDEX = 0

- 6. VERIFY (L1), Car\_Door\_Command = (Number of car doors present in the Lift), ARRAY INDEX = 0
- 7. VERIFY (L1), Landing\_Door\_Status = (Number of car doors present in the Lift), ARRAY INDEX = 0
- 8. CHECK (Array index 0 of all these properties shall be same)

#### 7.3.2.X47.1.12 Landing\_Door\_Status Tracks Car\_Door\_Status Test

Purpose: To verify that the status of Car\_Door\_Status property of lift is as same as that of the Landing\_Door\_Status property at a particular floor.

Test Concept: Car\_Door\_Status property of Lift (L1) object is subjected for different BACnetDoorStatus provided by changing the door status of real time lift connected to IUT or writing to it. The door side and floor number of the lift is considered in this case. The Landing\_Door\_Status property value at a specified array index (door size) for a particular floor (where lift car is currently present) is checked to verify that it is same as that of the status provided to Car\_Door\_Status property. If Landing\_Door\_Status property is not present, then this test shall be skipped.

Configuration Requirements: For below steps 'Array Index' = (any valid value N;  $1 \le N \le$  number of doors of a car). Y = (any valid floor number of the lift connected to the IUT). Tester shall select any value X for Car\_Door\_Status supported by IUT.

Test Steps:

- IF (Car\_Door\_Status is writable) THEN WRITE (L1), Car\_Door\_Status = X, ARRAY INDEX = N ELSE
  - MAKE (Car\_Door\_Status = (Value of X), ARRAY INDEX = N)
- 2. VERIFY (L1), Car\_Door\_Status = X, ARRAY INDEX = N
- 3. VERIFY (L1), Car\_Position = Y,
- 4. VERIFY (L1), Landing\_Door\_Status = X, ARRAY INDEX = N
- 5. CHECK (Landing\_Door\_Status property value is X only for the Universal floor number Y)

#### 7.3.2.X47.1.13 Highest Universal floor number linking to Car\_Position and Next\_Stopping\_Floor properties

Purpose: This test verifies that the highest universal floor number of the Lift object can be the maximum value of above properties depending on the floor numbers

Test Concept: Lift Object (L1) Properties Car\_Position and Next\_Stopping\_Floor will be written with the value of highest universal floor number and greater. If there is a physical lift or any alternate way for changing the highest universal floor number, change and REPEAT all the steps else omit. If any of the dependable properties are not writable, then skip the specific property from the test.

This test shall be skipped if Floor\_Text property is not present.

Configuration Requirements: For below steps 'Property Value' = (Y = highest universal floor number of the lift connected to the IUT). If Next\_Stopping\_Floor property is not present, then respective steps shall be skipped.

- 1. VERIFY (L1), Floor\_Text = Y, ARRAY INDEX = 0
- 2. IF (Car\_Position is writable) THEN WRITE (L1), Car\_Position = Y VERIFY (L1), Car\_Position = Y
- 3. TRANSMIT WriteProperty-Request, 'Object Identifier' = (L1), 'Property Identifier' = Car\_Position, 'Property Value' = Y+1
- 4. RECEIVE BACnet-Error-PDU, 'Error Class' = PROPERTY,

'Error Code' = VALUE\_OUT\_OF\_RANGE

- 5. IF (Next\_Stopping\_Floor is writable) THEN WRITE (L1), Next\_Stopping\_Floor = Y VERIFY (L1), Next\_Stopping\_Floor = Y
- 6. TRANSMIT WriteProperty-Request, 'Object Identifier' = (L1), 'Property Identifier' = Next\_Stopping\_Floor, 'Property Value' = Y+1
- 7. RECEIVE BACnet-Error-PDU, 'Error Class' = PROPERTY, 'Error Code' = VALUE\_OUT\_OF\_RANGE

# 7.3.2.X47.1.14 Highest Universal floor number linking to Assigned\_Landing\_Calls, Making\_Car\_Call and Registered\_Car\_Call properties

Purpose: This test verifies that the highest universal floor number of the Lift object can be the maximum value of above properties depending on the floor numbers

Test Concept: Lift Object (L1) Properties Assigned\_Landing\_Calls, Making\_Car\_Call and Registered\_Car\_Call will be written with the value of highest universal floor number and greater. If there is a physical lift or any alternate way for changing the highest universal floor number, change and REPEAT all the steps else omit. If any of the dependable properties are not writable, then skip the specific property from the test. This test shall be skipped if Floor\_Text property is not present.

Configuration Requirements: For below steps 'Array Index' = (any valid value N;  $1 \le N \le$  number of doors of a car) and 'Property Value' = (Y = highest universal floor number of the lift). If any of the dependable properties are not writable, then MAKE Out\_Of\_Service TRUE and then write, else skip the specific property from the test.

- 1. VERIFY (L1), Floor\_Text = Y, ARRAY INDEX = 0
- 2. IF (Making\_Car\_Call is writable) THEN WRITE (L1), Making\_Car\_Call = Y, ARRAY INDEX = N VERIFY (L1), Making\_Car\_Call = Y, ARRAY INDEX = N,
- TRANSMIT WriteProperty-Request, 'Object Identifier' = (L1), 'Property Identifier' = Making\_Car\_Call, 'Property Value' = Y+1
- 4. RECEIVE BACnet-Error-PDU, 'Error Class' = PROPERTY, 'Error Code' = VALUE\_OUT\_OF\_RANGE
- 5. IF (Registered\_Car\_Call is writable) THEN WRITE (L1), Registered\_Car\_Call = Y, ARRAY INDEX = N
- VERIFY (L1), Registered\_Car\_Call = Y, ARRAY INDEX = N,
- 7. TRANSMIT WriteProperty-Request,
   'Object Identifier' = (L1),
   'Property Identifier' = Registered\_Car\_Call,
   'Property Value' = Y+1
- 8. RECEIVE BACnet-Error-PDU, 'Error Class' = PROPERTY, 'Error Code' = VALUE\_OUT\_OF\_RANGE
- 9. IF (Assigned\_Landing\_Call is writable) THEN WRITE (L1), Assigned\_Landing\_Call = (Y, at Z Direction), ARRAY INDEX = N
- 10. VERIFY (L1), Assigned\_Landing\_Call = (Y, at Z Direction), ARRAY INDEX = N
- 11. TRANSMIT WriteProperty-Request, 'Object Identifier' = (L1),

'Property Identifier' = Assigned\_Landing\_Call, 'Property Value' = (Y+1, at Z Direction)
12. RECEIVE BACnet-Error-PDU, 'Error Class' = PROPERTY, 'Error Code' = VALUE OUT OF RANGE

#### 7.3.2.X47.1.15 Energy\_Meter\_Ref Property Tests

Purpose: To verify linking of Energy\_Meter property and Energy\_Meter\_Ref property.

Test Concept: If the Energy\_Meter\_Ref property of Lift object (L1) is present and initialized (contains an instance other than 4194303), then the Energy\_Meter property, if present, shall have a value of 0.0. If Energy\_Meter\_Ref is present and is un-initialized, then the value of Energy\_Meter property shall have any valid value.

Test Steps:

 IF (Energy\_Meter\_Ref is present and initialized with instance other than 4194303) THEN VERIFY Energy\_Meter = 0.0 ELSE VERIFY Energy\_Meter = (Any Valid Value)

### 7.3.2.X47.1.16 Higher\_Deck and Lower\_Deck Tests

Purpose: To verify that the Higher\_Deck and Lower\_Deck property of the Lift Object is referencing the Lift object that refers the car deck above and below the car deck represented by this Lift object.

Test Concept: The IUT under test is configured to have a 3-deck lift having 3 Lift Objects. The Higher\_Deck and Lower\_Deck Property of the Lift object is then read to verify that it is representing the correct Lift Object instances. If there is no higher deck or lower deck, then the object instance shall be 4194303.

Configuration Requirements: The IUT under test is configured to have a 3-deck lift having 3 Lift Object instances: higher deck (L1), middle deck (L2) and lower deck (L3). If the IUT have 2 Deck lift having 2 Lift Objects, then the test steps shall be modified accordingly and executed.

Test Steps:

- 1. VERIFY (L1), Higher\_Deck = (OBJECT, 4194303),
- 2. VERIFY (L1), Lower\_Deck = (L2),
- 3. VERIFY (L2), Higher\_Deck = (L1),
- 4. VERIFY (L2), Lower\_Deck = (L3),
- 5. VERIFY (L3), Higher\_Deck = (L2),
- 6. VERIFY (L3), Lower\_Deck = (OBJECT, 4194303)

# 7.3.2.X47.1.17 Linking of Assigned\_Landing\_Calls property of Lift Object to Landing\_Calls property of Elevator Group

Purpose: To verify that the Landing\_Calls property of Elevator Group also represents the active calls present in the Assigned\_Landing\_Calls property of the Lift object.

Test Concept: An Elevator Group is available, supports Landing\_Calls property, and it contains at least one Lift object within this group. Assigned\_Landing\_Calls property of the Lift is updated with the Floor number and direction for the lift. Landing\_Calls property of the Elevator Group object shall have the value as per the Assigned\_Landing\_Calls represented by this Lift object. For implementations where it is not possible to write to Assigned\_Landing\_Calls, this test shall be skipped.
Configuration Requirements: The Lift (L1) should be present in the Group\_Members property of Elevator Group (EG1). Lowest universal floor number of the lift < A < Highest universal floor number of the lift. Lowest universal floor number of the lift <= X <= Highest universal floor number of the lift. B = (UP | DOWN | UP\_AND\_DOWN) and C = (B | UP\_AND\_DOWN).

Test Steps:

- 1. IF (Assigned\_Landing\_Calls is writable) THEN WRITE Assigned\_Landing\_Calls = (Floor Number A, Direction B)
- 2. VERIFY (L1), Assigned\_Landing\_Calls = (Floor Number A, Direction B)
- 3. VERIFY (EG1), Landing\_Calls = (Floor Number A, Direction C | Destination X)

Notes to Tester: Landing\_Calls property may contain other entries from same lift or different lifts connected under same Elevator Group.

# **BTL-TP15.0-0.5.0 Test Considerations for Network Port OPTIONAL properties clarified**

A device including a Network Port object and claiming Protocol\_Revision 18 or higher must comply with the following section.

[In BTL Test Plan sections, add indicated Directives to apply during the performance of existing BTL Specified tests 9.20.1.8 and 9.20.1.9]

**Reason for Change:** There are some properties that had Conformance code "Required" in Protocol\_Revision 17 Some properties in Network Port object that had Conformance code "Required" in Protocol\_Revision 17, in Protocol\_Revision 18 changed their Conformance code to "Optional". See <u>http://www.bacnet.org/Interpretations/IC135-2016-1.pdf</u> for details.

### 4.4 Data Sharing - ReadPropertyMultiple - B

### **4.4.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL - 9.20.1.8 - Reading OPTIONAL Properties		
Tes	t Method	Manual
Cor	nfiguration	As per BTL Specified Tests.
Tes	t Conditionality	This test must be executed
Test	t Directives	Note: in Protocol_Revision 18 some of the properties indicated in Network Port object in Protocol_Revision 17 were changed from Required to Optional, and shall be returned when OPTIONAL is used with ReadPropertyMultiple. They shall not be returned when REQUIRED is used with ReadPropertyMultiple.
Tes	ting Hints	The pre-tester <i>shall</i> should apply this test to every object type. If the set of properties differs between instances of the same object type in the IUT, each form of the object type <i>shall</i> should be tested.
Not	es & Results	
BTL - 9.2	0.1.9 - Reading REQUIRED Proper	ties
Tes	t Method	Manual
Cor	ifiguration	As per BTL Specified Tests.
Tes	t Conditionality	This test must be executed
Tes	t Directives	Note: in Protocol_Revision 18 some of the properties indicated in Network Port object in Protocol_Revision 17 were changed from Required to Optional, and shall be returned when OPTIONAL is used with ReadPropertyMultiple. They shall not be returned when REQUIRED is used with ReadPropertyMultiple.
Test	ting Hints	The pre-tester <i>shall</i> should apply this test to every object type. If the set of properties differs between instances of the same object type in the IUT, each form of the object type <i>shall</i> should be tested.

Notes & Results
-----------------

Excerpt of 135-2016-Errata-Summary

Errata 73) Table 12-71, p. 516,

The Network Port object properties Network\_Number, Network\_Number\_Quality, and APDU\_Length are only required if the protocol level is BACNET\_APPLICATION.

Property Datatype	Conformance Code	
Unsigned16	$\mathbf{R}^{1} O^{I, Ibis}$	
BACnetNetworkNumberQuality	$\mathbf{R} O^{1bis}$	
Unsigned	$\mathbf{R} O^{1bis}$	
	Property Datatype Unsigned16 BACnetNetworkNumberQuality Unsigned 	

### Table 12-71. Properties of the Network Port Object Type

<sup>1</sup> Required to be writable in routers, secure devices, and any other device

that requires knowledge of the network number for proper operation.

<sup>1bis</sup> Required if Protocol\_Level is BACNET\_APPLICATION.

Shall be present if, and only if, the object supports execution of any of the values of the Command property. If present, this property shall be writable.

•••• •••

# BTL-TP15.0-0.6.0 Test of Write-BDT-NAK to Write-BDT service

The operation and manipulation of Broadcast Distribution Tables in devices claiming Protocol\_Revision 17 or higher is performed through operations on a Network Port object for each supported port.

[In BTL Test Plan, add test to end of Base Requirements for BACnet/IP - Annex J - BBMD]

### 9.4 BACnet/IP - Annex J - BBMD

### **9.4.1 Base Requirements**

The IUT acts, or can be made to act, as a BBMD device.

These base requirements must be met by any IUT that claims to support the Annex J BACnet/IP BBMD functionality.

• • •		
BTL	BTL - 7.3.2.X43.4 - Write-BDT service is required to return Write-BDT-NAK	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed in all devices claiming Protocol_Revision >= 17.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

[In BTL Specified Tests, add new test]

#### 7.3.2.X43.4 Write-BDT service is required to return Write-BDT-NAK

Reason for Change: Clause J.4.4.2 mandates a change and that all devices claiming  $Protocol_Revision >= 17$ , shall behave in this changed way.

Purpose: To verify that any IUT with Protocol\_Revision claimed as 17 or higher, will return Write-Broadcast-Distribution-Table NAK to every Write-Broadcast-Distribution-Table request.

Configuration Requirements: If the Protocol\_Revision claimed is less than 17, this test shall be skipped.

Test Steps:

- 1. TRANSMIT Write-Broadcast-Distribution-Table
- 2. RECEIVE BVLC-Result,

'Result Code' = Write-Broadcast-Distribution-Table NAK

# **BTL-TP15.0-0.7.0 Test Considerations for the NM-BBMDC-B BIBB**

Devices claiming this BIBB shall comply with the following section. This BIBB was specified in Protocol\_Revision 17.

#### **Overview:**

Addendum 135-2012*al* added the NM-BBMDC-B BIBB. This document makes needed changes in the BTL Test Package to claim NM-BBMDC-B.

These changes are not contained in any SSPC proposal.

#### **Changes:**

[In BTL Checklist, add new Network Management - BACnet Broadcast Management Device Configuration -B section]

Support	Listing	Option	
Network Management - BACnet Broadcast Management Device Configuration - B		gement - BACnet Broadcast Management Device Configuration - B	
	R	Base Requirements	
	R	Supports Registration by Foreign Devices	
	BTL-C <sup>1</sup>	Executes Write-Broadcast-Distribution-Table	
	C <sup>2</sup> Supports configurable BBMD_Broadcast_Distribution_Table property		
<sup>1</sup> Th	<sup>1</sup> This option is required if the IUT claims Protocol_Revision 16 or lower.		
<sup>2</sup> Th	<sup>2</sup> This option is required if the IUT claims Protocol_Revision 17 or higher.		

[In BTL Test Plan, add new Network Management - BACnet Broadcast Management Device Configuration -B sections at end of section 10]

### 10.X3 Network Management - BACnet Broadcast Management Device Configuration - B

These tests are designed for testing implementations of a BACnet Broadcast Management Device, including the execution of Network Layer and Application Layer commands to configure the BBMD.

### **10.X3.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL	BTL - 14.2.1.2 - Execute Forwarded-NPDU (Two-hop Distribution)		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD	
		Functionality.	
	Test Directives		
	Testing Hints		

Notes & Results	
BTL - 14.2.2.2 - Execute O	riginal-Broadcast-NPDU (Two-hop Distribution)
Test Method	Manual
Configuration	As per <b>BTL Specified Tests</b> .
Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
<b>Test Directives</b>	
<b>Testing Hints</b>	
Notes & Results	
135.1-2013 - 14.2.3 - Execu	te Original-Unicast-NPDU
Test Method	Manual
Configuration	As per <b>BTL Specified Tests</b> .
Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
<b>Test Directives</b>	
Testing Hints	
Notes & Results	
	inal-Broadcast-NPDU Which Shall Be Forwarded (Two-hop Distribution)
Test Method	Manual
Configuration	As per ASHRAE 135.1-2013.
Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
Test Directives	
Testing Hints	
Notes & Results	
BTL - 14.7.1.2 - Broadcast	Message from Directly Connected IP Subnet (Two-hop Distribution)
Test Method	Manual
Configuration	As per <b>BTL Specified Tests</b> .
Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
Test Directives	
<b>Testing Hints</b>	
Notes & Results	
BTL - 14.7.2.2 - Broadcast	Message Forwarded by a Peer BBMD (Two-hop Distribution)
Test Method	Manual
Configuration	As per <b>BTL Specified Tests</b> .
Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
Test Directives	
Testing Hints	
Notes & Results	
	al-Broadcast-NPDU

	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
		Functionality.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### **10.X3.2 Supports Registration by Foreign Devices**

While configured as a BBMD, the IUT supports, or can be made to support, registration by Foreign Devices and forwards as original BACnet/IP unicasts to each, any broadcasts it processes.

BTL	FL - 14.X10.2 - Holds at least 5 Foreign Device Registrations		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 14.X10.3 - Negative <b>F</b>	Foreign Device Registration when FD_Supported is FALSE	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
135.1	-2013 - 14.6.1 - Execut	e Read-Foreign-Device-Table	
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD	
		Functionality.	
	Test Directives		
	Testing Hints		
	Notes & Results		
135.1	-2013 - 14.6.3.1 - Non-	zero-Duration Foreign Device Table Timer Operations	
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD	
		Functionality.	
	Test Directives		
	Testing Hints		
	Notes & Results		
135.1	-2013 - 14.6.5 - Execut	e Delete-Foreign-Device-Table-Entry Which Should Be Rejected	

	Test Method	Manual
	Configuration	As per BTL Specified Tests
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
	·	Functionality.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 14.6.6 - Execut	e Delete-Foreign-Device-Table-Entry
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
		Functionality.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 14.7.3.2 - Broadcast N	Message From a Foreign Device (Two-hop Distribution)
	Test Method	Manual
	Configuration	As per BTL Specified Tests
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
		Functionality.
	Test Directives	
	Testing Hints	
	Notes & Results	

### **10.X3.3 Executes Write-Broadcast-Distribution-Table**

The IUT executes Write-Broadcast-Distribution-Table to update the configured peer BBMDs.

135.1-2013 - 14.3.1 - Execute Write-Broadcast-Distribution-Table (Table Growth)		
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
		Functionality.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 14.3.2 - Execut	e Write-Broadcast-Distribution-Table (Table Shrinkage)
135.1	-2013 - 14.3.2 - Execut Test Method	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual
135.1	-2013 - 14.3.2 - Execut Test Method Configuration	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013.
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality Test Directives	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality Test Directives Testing Hints	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.
135.1	-2013 - 14.3.2 - Execut Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	e Write-Broadcast-Distribution-Table (Table Shrinkage) Manual As per ASHRAE 135.1-2013. This test may be skipped if the IUT claims support for BACnet/IP - BBMD Functionality.

Previous Session		
	Test Method	Manual
	Configuration	As per BTL Specified Tests
	Test Conditionality	This test may be skipped if the IUT claims support for BACnet/IP - BBMD
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 14.X10.1 - Broadcast	t-Distribution-Table Holds at least 5 Entries
	Test Method	Manual
	Configuration	As per BTL Specified Tests
	<b>Test Conditionality</b>	Must be executed.
	<b>Test Directives</b>	
	<b>Testing Hints</b>	
	Notes & Results	

# BTL - 14.3.3 - Verify Broadcast Distribution Table Created from the Configuration Saved During the

### 10.X3.4 Supports BBMD\_Broadcast\_Distribution\_Table property

The IUT supports the configurable BBMD\_Broadcast\_Distribution\_Table property in Network Port objects to configure peer BBMDs.

BTL - 14.X10.4 - BBMD_Broadcast_Distribution_Table Holds at Least 5 Entries		
	Test Method	Manual
	Configuration	As per BTL Specified Tests
	<b>Test Conditionality</b>	Must be executed.
	<b>Test Directives</b>	
	<b>Testing Hints</b>	
	Notes & Results	
BTL	- 7.3.2.X43.4 - Write-B	BDT service is required to return Write-BDT-NAK
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed in all devices claiming Protocol_Revision >= 17.
	<b>Test Directives</b>	
	<b>Testing Hints</b>	
	Notes & Results	

[Add in BTL Specified Tests, these four new tests]

### 14.X10.1 - Broadcast-Distribution-Table Holds at Least 5 Entries

Reason For Change: NM-BBMDC-B specifically mandates this capacity behavior is supported by the product.

Purpose: Verify that IUT implements capacity mandated for the product by NM-BBMDC-B.

Test Concept: Fill the Broadcast Distribution Table with at least five distinct peer BBMDs entries (in addition to the entry containing the address of itself in the table).

Configuration Requirements: In a device claiming Protocol\_Revision 16 or less, the means by which the product's Broadcast-Distribution-Table is configured is not restricted to BACnet network transmissions, and can be through the product's end-user interface.

Test Steps:

- 1. MAKE (IUT enter mode functioning as a BBMD implementation)
- 2. MAKE Broadcast\_Distribution\_Table = (its own entry and entries for at least 5 other BBMDs))
- 3. TRANSMIT Read- Broadcast-Distribution-Table
- 4. RECEIVE Read-Broadcast-Distribution-Table-Ack, 'List of BDT Entries' = (the table as configured, in any order)

#### 14.X10.2 - Holds at Least 5 Foreign Device Registrations

Reason For Change: NM-BBMDC-B specifically mandates this capacity behavior is supported by BBMDs.

Purpose: Verify that when configured to accept foreign device registrations, the IUT supports at least five simultaneous foreign device registrations.

Test Concept: The IUT is configured to support foreign device registrations. Five Register-Foreign-Device requests are sent from 5 different devices, to verify that it supports five registrations simultaneously in the FDT.

Configuration Requirements: Set BBMD\_Accept\_FD\_Registrations in the Network Port object representing the port operating as a BBMD to TRUE. The TD will be configured to emulate 5 devices.

Test Steps:

```
1. REPEAT X = 1 to 5 {
    TRANSMIT Register-Foreign-Device
    SOURCE = (device X)
        'Time-to-Live' = (a value longer than the length of the test)
    RECEIVE BVLC-Result,
        'Result Code' = Successful completion
}
```

#### 14.X10.3 - Negative Foreign Device Registration when FD\_Supported is FALSE

Reason For Change: The standard specifically mandates that BBMD\_Accept\_FD\_Registrations property is writable if present in BBMDs.

Purpose: Verify that when BBMD\_Accept\_FD\_Registrations is configured as FALSE, the BBMD will accept no more foreign device registrations.

Test Concept: The IUT is configured with BBMD\_Accept\_FD\_Registrations property as FALSE. Then it is verified that no more Register-Foreign-Device registrations succeed, though those already in the FDT operate as normal.

Configuration Requirements: BBMD\_Accept\_FD\_Registrations in the Network Port object representing the port is initially TRUE. If no Network Port object contains the BBMD\_Accept\_FD\_Registrations property, then this test shall be skipped.

Test Steps:

- 1. WRITE BBMD\_Accept\_FD\_Registrations = FALSE
- 2. TRANSMIT Register-Foreign-Device
- 3. RECEIVE BVLC-Result,

'Result Code' = Register-Foreign-Device NAK

#### 14.X10.4 - BBMD\_Broadcast\_Distribution\_Table Holds at Least 5 Entries

Reason For Change: NM-BBMDC-B specifically mandates this capacity behavior is supported by the product.

Purpose: Verify that the IUT supports at least 5 peer BBMD entries in its broadcast distribution table.

Test Concept: Fill the BBMD\_Broadcast\_Distribution\_Table with at least five distinct peer BBMDs entries (in addition to the entry containing the address of itself in the table).

Configuration Requirements: the IUT is configured to operate as a BBMD.

Test Steps:

- 1. WRITE BBMD\_Broadcast\_Distribution\_Table = (its own entry and entries for at least 5 other BBMDs)
- 2. MAKE (that configuration active)
- 3. TRANSMIT Read- Broadcast-Distribution-Table
- 4. RECEIVE Read-Broadcast-Distribution-Table-Ack, 'List of BDT Entries' = (the table as configured, in any order)

# BTL-TP15.0-1.1.0 Tests for the FAULT\_LISTED algorithm

Devices claiming support for CHANGE\_OF\_RELIABILITY with FAULT\_LISTED algorithm must claim Protocol\_Revision 18 and comply with the following section.

#### **Overview:**

Addendum 135-2012*aq*-3 at Protocol\_Revision 18 added new FAULT\_LISTED algorithm to vertical transport objects that provide fault reporting, and to the Event Enrollment object.

#### **Changes:**

[In BTL Specified Tests, add a new test] 8.4.X1 CHANGE\_OF\_RELIABILITY Tests (ConfirmedEventNotification)

#### 8.4.X1.13 Change\_Of\_Reliability with FAULT\_LISTED Algorithm (ConfirmedEventNotification)

Purpose: To verify the correct operation of the FAULT\_LISTED event algorithm.

Test Concept: Select a fault detecting object O1 which is configured to use the FAULT\_LISTED algorithm. Ensure that no other fault conditions exist in the object. Set pMonitoredList to FV1, any value whose presence in the list property indicates a FAULT\_LISTED fault condition. Verify the correct transition is generated. The fault condition is removed by setting pMonitoredList to empty, a value which indicates NO\_FAULT\_DETECTED and verify the correct transition is generated.

Configuration Requirements: O1 is configured to detect faults and to report those using confirmed event notifications. O1 is initially configured to have no fault conditions present, and has an Event\_State of NORMAL. FV1 is a value for pMonitoredList which indicates a fault condition, and an empty pMonitoredList does not indicate a fault condition.

Test Steps:

- 1. VERIFY pCurrentReliability = NO\_FAULT\_DETECTED
- 2. VERIFY Event\_State = NORMAL
- 3. IF (pMonitoredList is writable) THEN WRITE pMonitoredList = FV1
  - ELSE
  - MAKE (pMonitoredList = FV1)
- 4. BEFORE Notification Fail Time
  - RECEIVE ConfirmedEventNotification-Request,

	1 /
'Process Identifier' =	(any valid process Identifier),
'Initiating Device Identifier' =	IUT
'Event Object Identifier' =	01
'Time Stamp' =	(the current local time or sequence number),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	CHANGE_OF_RELIABILITY,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	ALARM   EVENT,
'AckRequired' =	TRUE   FALSE,
'From State' =	NORMAL,
'To State' =	FAULT,
'Event Values' =	( FAULT_LISTED,
	(T, T, ? ?),

(A list of valid values for properties required to be reported for O1, and 0 or more other properties of O1)

- 5. TRANSMIT BACnet-SimpleACK-PDU
- 6. VERIFY pCurrentReliability = FAULTS LISTED
- 7. VERIFY Event State = FAULT
- 8. IF (pMonitoredList is writable) THEN
- WRITE pMonitoredList = { }

ELSE

MAKE (pMonitoredList = { })

9. BEFORE Notification Fail Time

**RECEIVE** Co

quest,
(any valid process Identifier),
IUT
01
(the current local time or sequence number),
(the notification class configured for O1),
(the value configured for the transition),
CHANGE_OF_RELIABILITY,
(optional, any valid message text),
ALARM   EVENT,
TRUE   FALSE,
FAULT,
NORMAL,
( NO_FAULT_DETECTED,
(F, F, ? ?),
(A list of valid values for properties required to be reported
for O1, and 0 or more other properties of O1)

)

)

10. TRANSMIT BACnet-SimpleACK-PDU

11. pCurrentReliability = NO\_FAULT\_DETECTED

12. VERIFY Event\_State = NORMAL

[In BTL Specified Tests, add a new test in this section] 8.5.X1 CHANGE\_OF\_RELIABILITY Tests

#### 8.5.X1.14 Change Of Reliability with FAULT LISTED Algorithm (UnconfirmedEventNotification)

Purpose: To verify the correct operation of the FAULT LISTED event algorithm.

Test Concept: Select a fault detecting object O1 which is configured to use the FAULT LISTED algorithm. Ensure that no other fault conditions exist in the object. Set pMonitoredList to FV1, any value whose presence in the list property indicates a FAULT\_LISTED fault condition. Verify the correct transition is generated. The fault condition is removed by setting pMonitoredList to empty which indicates NO FAULT DETECTED and verify the correct transition is generated.

Configuration Requirements: O1 is configured to detect faults and to report those using unconfirmed event notifications. O1 is initially configured to have no fault conditions present, and has an Event\_State of NORMAL. FV1 is a value for pMonitoredList which indicates a fault condition, and an empty pMonitoredList does not indicate a fault condition.

Test Steps: The test steps for this test case are identical to the test steps in 'Change Of Reliability with FAULT LISTED Algorithm (ConfirmedEventNotification)' except that the ConfirmedEventNotification requests are UnconfirmedEventNotification requests and the TD does not acknowledge receiving the notifications.

# BTL-TP15.0-1.2.0 Tests for FAULT-to-FAULT transitions in FAULT\_LISTED algorithm

Devices claiming support for FAULT-to-FAULT transitions in the FAULT\_LISTED algorithm must claim support for Protocol\_Revision 18 and comply with the following section.

### **Overview:**

Addendum 135-2012*aq*-3 at Protocol\_Revision 18 the added FAULT\_LISTED algorithm for vertical transport objects provides for optional fault-to-fault reporting.

### **Changes:**

[In BTL Checklist, add a new optional lineitem under Escalator section in existing 3. Object testing.]

Support	Listing	Option	
Esca	Escalator Object		
	0	Supports FAULT-to-FAULT transitions in FAULT_LISTED	

[In BTL Test Plan, add an additional section under Escalator in order to optionally execute the testing in 3.X46.7 as indicated.]

### 3.X46 Escalator Object

### **3.X46.7 Supports FAULT-to-FAULT transitions in FAULT\_LISTED**

These requirements must be met by any IUT that can contain more than one element or different values in the Fault\_Signals property in any of its Escalator objects.

BTL - 8.5.X1.15 - Change_Of_Reliability FAULT-to-FAULT transitions in FAULT_LISTED		
Test Method Manual		Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

[In BTL Specified Tests, add a new test in this section]

### 8.5.X1 CHANGE\_OF\_RELIABILITY Tests

#### 8.5.X1.15 Change\_Of\_Reliability FAULT-to-FAULT transitions in FAULT\_LISTED

Purpose: To verify the correct operation of FAULT-to-FAULT transitions in FAULT\_LISTED event algorithm.

Test Concept: Select a fault detecting object O1 which is configured to use the FAULT\_LISTED algorithm. Ensure that a fault condition exists in the object. Set pMonitoredList to FV1, any set of non-empty values different from the current set of values. Verify the correct transition is generated. The fault condition is removed by setting

pMonitoredList to empty, a value which indicates NO\_FAULT\_DETECTED and verify the correct transition is generated.

Configuration Requirements: O1 is configured to detect faults and to report those using unconfirmed event notifications. O1 is initially configured to have a fault conditions present by pMonitoredList containing a non-empty value, and has an Event\_State of FAULT. FV1 is a value or set of values for pMonitoredList, and which the IUT will support in the pMonitoredList value. An empty pMonitoredList does not indicate a fault condition.

Test Steps:

- 1. VERIFY pCurrentReliability = FAULTS\_LISTED
- 2. VERIFY Event\_State = FAULT
- 3. IF (pMonitoredList is writable) THEN
  - WRITE pMonitoredList = FV1

ELSE

- MAKE (pMonitoredList = FV1)
- 4. BEFORE Notification Fail Time

RECEIVE UnconfirmedEventNotification-Request,

'Process Identifier' = (any valid process Identifier), 'Initiating Device Identifier' = IUT 'Event Object Identifier' = 01 'Time Stamp' = (the current local time or sequence number), 'Notification Class' = (the notification class configured for O1), 'Priority' = (the value configured for the transition), 'Event Type' = CHANGE\_OF\_RELIABILITY, 'Message Text' = (optional, any valid message text), 'Notify Type' = ALARM | EVENT, 'AckRequired' = TRUE | FALSE, 'From State' = FAULT, 'To State' = FAULT, 'Event Values' = (FAULT\_LISTED, (T, T, ??), (A list of valid values for properties required to be reported

for O1, and 0 or more other properties of O1)

)

- 5. VERIFY pCurrentReliability = FAULTS\_LISTED
- 6. VERIFY Event\_State = FAULT
- 7. IF (pMonitoredList is writable) THEN WRITE pMonitoredList = {}

ELSE

- MAKE (pMonitoredList = { })
- 8. BEFORE Notification Fail Time

RECEIVE UnconfirmedEventNotification-Request,

'Process Identifier' =	(any valid process Identifier),
'Initiating Device Identifier' =	IUT
'Event Object Identifier' =	01
'Time Stamp' =	(the current local time or sequence number),
'Notification Class' =	(the notification class configured for O1),
'Priority' =	(the value configured for the transition),
'Event Type' =	CHANGE_OF_RELIABILITY,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	ALARM   EVENT,
'AckRequired' =	TRUE   FALSE,
'From State' =	FAULT,
'To State' =	NORMAL,
'Event Values' =	( NO_FAULT_DETECTED,

(F, F, ? ?), (A list of valid values for properties required to be reported for O1, and 0 or more other properties of O1)

9. VERIFY pCurrentReliability = NO\_FAULT\_DETECTED10. VERIFY Event\_State = NORMAL

)

# **BTL-TP15.0-2.1.0:** Binary Lighting Output object

Devices claiming support for a Binary Lighting Output object must claim support for Protocol\_Revision 16 and comply with the following section.

#### **Overview:**

Addendum 135-2012*az* added the Binary Lighting Output object. This document makes needed changes in the BTL Test Package to claim Binary Lighting Output object.

These changes are not contained in any SSPC proposal.

[In BTL Checklist, add Binary Lighting Output object type to Section 3, Objects]

Support	Listing	Option
Bin	arv Lighting	9 Output Object
2111	R	Base Requirements
	R	Supports command prioritization
	S	Supports writable Out_Of_Service properties
	0	Supports blink-warn
	0	Supports writable Polarity property
	0	Supports strike count tracking
	0	Supports elapsed active time tracking
	0	Contains an object with Reliability_Evaluation_Inhibit Property

[In BTL Test Plan, add Binary Lighting Output object tests in section 3.X41. In the following addition of new clauses of the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold-italic** as shown.]

### 3.X41 Binary Lighting Output Object

### **3.X41.1 Base Requirements**

Base requirements must be met by any IUT that can contain Binary Lighting Output objects. All requirements for this object are specified in other sections.

### **3.X41.2 Supports Command Prioritization**

135.1	35.1-2013 - 7.3.1.2 - Relinquish Default Test		
Test Method Manual		Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	If no object can be made to meet the configuration requirements, this test	
		shall be skipped.	
Test Directives			
	<b>Testing Hints</b>		

	Notes & Results	
135.1	-2013 - 7.3.1.3 - Comm	and Prioritization Test
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	Must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### 3.X41.3 Supports Writable Out\_Of\_Service Properties

The Out\_Of\_Service property in Binary Lighting Output objects contained in the IUT are writable.

35.1-2013 - 7.3.1.1 - Out_Of_Service, Status_Flags, and Reliability Tests		
	Test Method	Manual
	Configuration	This test shall be executed using a Binary Lighting Output object.
	<b>Test Conditionality</b>	If Out_Of_Service is writable, this test must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### 3.X41.4 Supports Blink-warn

The IUT supports blink-warn the Binary Output object.

BTL -	ГL - 7.3.1.X.1 - Blink Warn WARN Command Test			
	Test Method	st Method Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	Must be executed.		
	<b>Test Directives</b>			
	Testing Hints			
	Notes & Results			
BTL -	- 7.3.1.X.2 - Blink War	n WARN_OFF Command Test		
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	Must be executed.		
	Test Directives			
	Testing Hints			
	Notes & Results			
BTL-	BTL - 7.3.1.X.3 - Blink Warn WARN_RELINQUISH Command Test			
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	Must be executed.		
	<b>Test Directives</b>			
	<b>Testing Hints</b>			

	Notes & Results		
BTL	L - 7.3.1.X.4 - Blink Warn STOP Command Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 7.3.1.X.5 - Blink War	n WARN Command Failure Test	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives	Repeat the test with WARN_OFF and WARN_RELINQUISH commands	
	Testing Hints		
	Notes & Results		
DTI	721V6 Dlink Wan	n WADN OFF Command Failure Test	
DIL	- 7.3.1.A.0 - DIIIK War Tost Mothod	Menuel	
	Configuration	Manual	
	Tost Conditionality	As per <b>BIL specified Tests</b> .	
	Test Conditionality	Must be executed.	
	Test Directives		
	Notes & Results		
	Tioles & Results		
BTL ·	- 7.3.1.X.7 - Blink War	n WARN RELINOUISH Command Failure Test	
	Test Method	Manual	
	Configuration	As per <b>BTL Specified Tests</b> .	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL ·	- 7.3.1.X.8 - Blink War	n WARN_OFF Command Halted Test	
	Test Method	Manual	
	Configuration	As per <b>BTL Specified Tests</b> .	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
рті	721V0 DE-L-W	NADN DELINQUISH Command Halted Test	
BIL	- 1.J.I.A.Y - Billik War	II WARIA_KELIINQUISH COMMANA HAItea Test	
	Configuration	Manual	
	Test Conditionality	As per DIL Specifica Lesis. Must be executed	
	Test Directives		
	Testing Hints		
1			

Notes & Results	

### **3.X41.5** Supports writable Polarity property

The IUT supports a writable Polarity property in the Binary Output object.

135.1	35.1-2013 - 7.3.2.6.3 - Polarity Property Tests		
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	Must be executed	
	<b>Test Directives</b>		
	<b>Testing Hints</b>		
	Notes & Results		

### 3.X41.6 Supports Strike Count Tracking

The properties of the Binary Lighting Output object that collectively tracks strike counts as required.

BTL -	TL - 7.3.2.X41.10 - Binary Lighting Output Object Strike Count Tests		
	Test Method Manual		
	Configuration As per BTL Specified Tests.		
	Test Conditionality	Aust be executed if Strike_Count property supported.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X41.7 Supports Elapsed Active Time Tracking

The properties of binary objects that collectively track active time function as required.

BTL	TL - 7.3.1.9 - Binary Object Elapsed Active Time Tests		
	Test Method	Manual	
	Configuration	As per <b>BTL Specified Tests</b> .	
	<b>Test Conditionality</b>	If all of the active time properties are supported, it must be executed.	
	<b>Test Directives</b>		
	<b>Testing Hints</b>		
	Notes & Results		

### 3.X41.8 Contains an object with Reliability\_Evaluation\_Inhibit Property

The IUT contains, or can be made to contain, a Reliability\_Evaluation\_Inhibit property that is configurable to a value of TRUE.

<b>3TL</b>	- 7.3.1.X8.1 - Reliability	_Evaluation_Inhibit Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If no object exists in the IUT for which fault conditions can be generated,
		then this test shall be skipped.
	Test Directives	

	Testing Hints		
	Notes & Results		
BTL	- 7.3.1.X8.2 - Reliability	_Evaluation_Inhibit Summarization Test	
	Test Method	Manual	
	Configuration As per <i>BTL Specified Tests</i> .		
	<b>Test Conditionality</b> If no object exists in the IUT for which fault conditions can be		
		generated, then this test shall be skipped.	
	Test Directives	- · · · · · · · · · · · · · · · · · · ·	
	Testing Hints		
	Notes & Results		

[In BTL Specified Tests, add non-object specific tests for Blink in section 7.3.1.X, applicable to both Lighting Output or Binary Lighting Output objects.]

#### 7.3.1.X.1 Blink Warn WARN Command Test

Purpose: To verify the correct operation of the blink-warn WARN command.

Test Concept: Select an object O1 that supports blink-warn WARN command. Ensure O1 is not in egress mode and the specific properties have been configured to support blink-warn. Execute blink-warn WARN command by writing C1 to PROP\_REF at a priority PTY1 of O1 and validate the specified blink-warn command functions correctly. Validate the Priority\_Array value at priority PTY1 remains.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. The Priority\_Array at PTY1 has a value V1, Blink\_Warn\_Enable is TRUE, Egress\_Active is FALSE.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN	-1.0 if PROP_REF = Present_Value, otherwise WARN
V1	ON	>1.0

Test Steps:

- 1. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 2. VERIFY Blink\_Warn\_Enable = TRUE
- 3. VERIFY Egress\_Active = FALSE
- 4. WRITE  $PROP\_REF = C1$ , PRIORITY = PTY1
- 5. BEFORE Internal Processing Fail Time CHECK (blink-warn occurred)
- 6. VERIFY Egress\_Active = FALSE
- 7. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1

#### 7.3.1.X.2 Blink Warn WARN\_OFF Command Test

Purpose: To verify the correct operation of the blink-warn WARN\_OFF command.

Test Concept: Select an object O1 that supports blink-warn commands. Ensure O1 is not in egress mode and the specific properties have been configured to support blink Warn. Execute blink-warn WARN\_OFF command by

writing C1 to PROP\_REF at a priority PTY1 of O1 and validate the specified blink-warn command functions correctly. Validate the Priority\_Array value at priority PTY1 after Egress\_Time seconds has elapsed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. The Priority\_Array at PTY1 has a value V1, Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value, Egress\_Active is FALSE, and Egress\_Time is a non-zero value..

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_OFF	-3.0 if PROP_REF = Present_Value, otherwise WARN_OFF
V1	ON	>1.0
V2	OFF	0.0

Test Steps:

- 1. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 2. VERIFY Blink\_Warn\_Enable = TRUE
- 3. VERIFY Egress\_Time > 0
- 4. VERIFY Egress\_Active = FALSE
- 5. WRITE  $PROP_REF = C1$ , PRIORITY = PTY1
- 6. T1 = current local time
- 7. BEFORE Internal Processing Fail Time CHECK (blink-warn occurred)
- 8. WHILE (Egress\_Active = TRUE) VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 9. T2 = current local time
- 10. VERIFY (T1 T2) ~= Egress\_Time +/- Internal Processing Fail Time
- 11. VERIFY Priority\_ Array = V2, ARRAY\_INDEX = PTY1

### 7.3.1.X.3 Blink Warn WARN\_RELINQUISH Command Test

Purpose: To verify the correct operation of the Blink Warn WARN\_RELINQUISH commands.

Test Concept: Select an object O1 that supports blink-warn commands. Ensure O1 is not in egress mode and the specific properties have been configured to support blink-warn. Execute blink-warn WARN\_RELINQUISH command by writing C1 to PROP\_REF at a priority PTY1 of O1 and validate the specified blink-warn command functions correctly. Validate the Priority\_Array value at priority PTY1 after Egress\_Time seconds has elapsed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL, slots numerically greater than PTY1 shall be V0 and no internal algorithms are issuing commands to O1 at any priority. The Priority\_Array at PTY1 has a value V1, Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value, Egress\_Active is FALSE, and Relinquish\_Default has a value, V2.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_RELINQUISH	-2.0 if PROP_REF = Present_Value, otherwise WARN_OFF
V0	NULL or OFF	NULL or 0.0
V1	ON	>1.0
V2	ON	>= 1.0 and < V1

Test Steps:

- 1. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 2. VERIFY Blink\_Warn\_Enable = TRUE

- VERIFY Egress\_Time > 0 3.
- 4. VERIFY Egress Active = FALSE
- 5. WRITE PROP REF = C1, PRIORITY = PTY1
- T1 = current local time6.
- 7. **BEFORE Internal Processing Fail Time** CHECK (blink-warn occurred)
- 8. WHILE (Egress\_Active = TRUE) VERIFY Priority Array = V1, ARRAY INDEX = PTY1
- 9. T2 = current local time
- 10. VERIFY  $(T1 T2) \sim =$  Egress\_Time +/- Internal Processing Fail Time
- 11. VERIFY Priority Array = NULL, ARRAY INDEX = PTY1

#### 7.3.1.X.4 Blink Warn STOP Command Test

Purpose: To verify the correct operation of the blink-warn STOP command.

Test Concept: Select an object O1 that supports blink-warn commands. Ensure O1 is not in egress mode and the specific properties have been configured to support blink-warn. Execute blink-warn command by writing C1 to PROP REF at a priority PTY1 of O1 and validate that blink-warn occurs. Before the Egress Time times out, STOP the egress process and validate the Priority\_Array value at PTY1 remains equal to V1 after Egress\_Time.

Configuration Requirements: O1 shall be configured such that all slots in the Priority Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. The Priority Array at PTY1 has a value V1. Blink Warn Enable is TRUE, Egress Time is a non-zero value, and Egress\_Active is FALSE.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Lighting_Command
C1	WARN_RELINQUISH or	WARN_RELINQUISH or WARN_OFF
	WARN_OFF	
V1	ON	>1.0

Test Steps:

- 1. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 2. VERIFY Blink\_Warn\_Enable = TRUE
- VERIFY Egress Time > 03.
- VERIFY Egress Active = FALSE 4.
- WRITE PROP\_REF = C1, PRIORITY = PTY1 5.
- T1 = current local time6.
- **BEFORE Internal Processing Fail Time** 7. CHECK (blink-warn occurred)
- VERIFY Egress Active = TRUE 8.
- WAIT less than Egress\_Time 9.
  - WRITE PROP\_REF = STOP, PRIORITY = PTY1
- 10. T2 = current local time
- 11. WAIT Internal Processing Fail Time
- VERIFY Egress\_Active = FALSE
   WAIT Egress\_Time (T2 T1) + Internal Processing Fail Time
- 14. VERIFY Priority Array = V1, ARRAY INDEX = PTY1

#### 7.3.1.X.5 Blink Warn WARN Command Failure Test

Purpose: To verify blink-warn WARN command does not occur when, the specified priority is not the highest active priority, the value at the specified priority is off or Blink\_Warn\_Enable is FALSE.

Test Concept: Select an object O1 that supports blink-warn commands. Configure O1 such that a blink-warn command would generate a blink-warn except set the specified failure conditions. Verify blink-warn does not occur and the Priority\_Array is not affected.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. Select a priority, PTY2, which is numerically less than PTY1 and not equal to 6. Blink\_Warn\_Enable is TRUE, Egress\_Active is FALSE.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN	-1.0 if PROP_REF = Present_Value, otherwise WARN
V1, V2	ON	>1.0
V3	OFF	0.0

Test Steps:

-- Test for the specified priority is not the highest active priority

- 1. VERIFY Blink\_Warn\_Enable = TRUE
- 2. WRITE Present\_Value = V1, PRIORITY = PTY1
- 3. VERIFY Egress\_Active = FALSE
- 4. WRITE Present\_Value = V2, PRIORITY = PTY2
- 5. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 6. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 7. VERIFY Egress\_Active = FALSE
- 8. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 9. WRITE Present\_Value = NULL, PRIORITY = PTY2

-- Test for the value at the specified priority is either OFF or 0.0

- 10. WRITE Present\_Value = V3, PRIORITY = PTY1
- 11. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 12. WAIT Internal Processing Fail Time
  - CHECK (blink-warn did not occur)
- 13. VERIFY Egress\_Active = FALSE
- 14. VERIFY Priority\_ Array = V3, ARRAY\_INDEX = PTY1
- 15. WRITE Present\_Value = V1, PRIORITY = PTY1

-- Test for Blink\_Warn\_Enable is FALSE

- 16. MAKE Blink\_Warn\_Enable = FALSE
- 17. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 18. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 19. VERIFY Egress\_Active = FALSE
- 20. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1

#### 7.3.1.X.6 Blink Warn WARN\_OFF Command Failure Test

Purpose: To verify blink-warn WARN\_OFF command does not occur when the specified priority is not the highest active priority, the Present\_Value is either 0.0 or OFF, or Blink\_Warn\_Enable is FALSE.

Test Concept: Select an object O1 that supports blink-warn commands. Configure O1 such that a blink-warn command would generate a blink-warn except set the specified failure conditions. Verify blink-warn does not occur and the Priority\_Array is correctly changed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value and Egress\_Active is FALSE.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_OFF	-3.0 if PROP_REF = Present_Value, otherwise WARN_OFF
V1, V2	ON	>1.0
V3	OFF	0.0

Test Steps:

-- Test for the specified priority is not the highest active priority

- 1. VERIFY Blink\_Warn\_Enable = TRUE
- 2. VERIFY Egress\_Time > 0
- 3. WRITE Present\_Value = V1, PRIORITY = PTY1
- 4. VERIFY Egress\_Active = FALSE
- 5. WRITE Present\_Value = V2, PRIORITY = PTY2, a value not equal to 6 and less than PTY1
- 6. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 7. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 8. VERIFY Egress\_Active = FALSE
- 9. VERIFY Priority\_ Array = V3 , ARRAY\_INDEX = PTY1
- 10. WRITE Present\_Value = V1, PRIORITY = PTY1

-- Test for the Present\_Value is OFF or 0.0

- 11. WRITE Present\_Value = V3, PRIORITY = PTY2, a value not equal to 6 and less than PTY1
- 12. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 13. WAIT **Internal Processing Fail Time** CHECK (blink-warn did not occur)
- 14. VERIFY Egress\_Active = FALSE
- 15. VERIFY Priority\_ Array = V3, ARRAY\_INDEX = PTY1
- 16. WRITE Present\_Value = NULL, PRIORITY = PTY2
- 17. WRITE Present\_Value = V1, PRIORITY = PTY1

-- Test for Blink\_Warn\_Enable is FALSE

- 18. MAKE Blink\_Warn\_Enable = FALSE
- 19. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 20. WAIT Internal Processing Fail Time
- CHECK (blink-warn did not occur)
- 21. VERIFY Egress\_Active = FALSE
- 22. VERIFY Priority\_ Array = V3, ARRAY\_INDEX = PTY1

#### 7.3.1.X.7 Blink Warn WARN\_RELINQUISH Command Failure Test

Purpose: To verify blink-warn WARN\_RELINQUISH command does not occur when the specified priority is not the highest active priority, the value at the specified priority is V0, the value of the next highest non-NULL priority, including Relinquish\_Default, is not V0, or Blink\_Warn\_Enable is FALSE.

Test Concept: Select an object O1 that supports blink-warn commands. Configure O1 such that a blink-warn command would generate a blink-warn except set the specified failure conditions. Verify blink-warn does not occur and the Priority\_Array is correctly changed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL, slots numerically greater than PTY1 shall be V0 and no internal algorithms are issuing

commands to O1 at any priority. Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value, Egress\_Active is FALSE and Relinquish\_Default is V0.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_RELINQUISH	-2.0 if PROP_REF = Present_Value, otherwise
		WARN_RELINQUISH
V0	OFF or NULL	0.0 or NULL
V1 to V5	ON	>1.0

Test Steps:

- -- Test for the specified priority is not the highest active priority
- 1. VERIFY Blink\_Warn\_Enable = TRUE
- 2. VERIFY Relinquish\_Default <> 0
- 3. VERIFY Egress\_Time > 0
- 4. WRITE Present\_Value = V1, PRIORITY = PTY1
- 5. VERIFY Egress\_Active = FALSE
- 6. WRITE Present\_Value = V2, PRIORITY = PTY2, a value not equal to 6 and less than PTY1
- 7. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 8. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 9. VERIFY Egress\_Active = FALSE
- 10. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1
- 11. WRITE Present\_Value = NULL, PRIORITY = PTY2

#### -- Test for the value at the specified priority is OFF or 0.0

- 12. WRITE Present\_Value = V6, PRIORITY = PTY1
- 13. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 14. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 15. VERIFY Egress Active = FALSE
- 16. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1

-- Test for the value at the specified priority is NULL

- 17. WRITE Present\_Value = NULL, PRIORITY = PTY1
- 18. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 19. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 20. VERIFY Egress\_Active = FALSE
- 21. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1

-- Test for the value of the next highest non-NULL priority is neither OFF or 0.0

- 22. WRITE Present\_Value = V1 PRIORITY = PTY1
- 23. WRITE Present\_Value = V3, PRIORITY = PTY3, a value numerically greater than PTY1
- 24. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 25. WAIT Internal Processing Fail Time
  - CHECK (blink-warn did not occur)
- 26. VERIFY Egress\_Active = FALSE
- 27. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1
- 28. WRITE Present\_Value = NULL, PRIORITY = PTY3

-- Test for the value of Relinquish\_Default is neither OFF or 0.0

- 29. WRITE Present\_Value = V1, PRIORITY = PTY1
- 30. WRITE Relinquish\_Default = V4

- 31. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 32. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 33. VERIFY Egress\_Active = FALSE
- 34. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1

-- Test for Blink\_Warn\_Enable is FALSE

- 35. WRITE Relinquish\_Default = V5
- 36. WRITE Present\_Value = V1, PRIORITY = PTY1
- 37. WRITE Blink\_Warn\_Enable = FALSE
- 38. WRITE PROP\_REFPresent\_Value = C1, PRIORITY = PTY1
- 39. WAIT Internal Processing Fail Time CHECK (blink-warn did not occur)
- 40. VERIFY Egress\_Active = FALSE
- 41. VERIFY Priority\_ Array = NULL, ARRAY\_INDEX = PTY1

### 7.3.1.X.8 Blink Warn WARN\_OFF Command Halted Test

Purpose: To verify blink-warn WARN\_OFF execution is halted when a higher priority entry is written or the Present\_Value at the specified priority is changed.

Test Concept: Select an object O1 that supports blink-warn commands. Configure O1 such that a blink-warn command will generate a blink-warn. Before the Egress timer expires, verify the specified actions clear the blink-warn properties and the Priority\_Array is correctly changed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value and Egress\_Active is FALSE.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_OFF	-3.0 if PROP_REF = Present_Value, otherwise WARN_OFF
V1 to V3	ON	>1.0
V4	OFF	0.0

Test Steps:

- -- Test for a higher priority entry is written to a non NULL value
- 1. WRITE Present\_Value = V1, PRIORITY = PTY1
- 2. VERIFY Blink\_Warn\_Enable = TRUE
- 3. VERIFY Egress\_Time > 0
- 4. VERIFY Egress\_Active = FALSE
- 5. WRITE PROP\_REF = C1, PRIORITY = PTY1
- 6. BEFORE Internal Processing Fail Time
  - CHECK (blink-warn occurred)
- 7. BEFORE Egress\_Active = FALSE

### WRITE Present\_Value = V2, PRIORITY = PTY2, a value not equal to 6 and less than PTY1

- 8. VERIFY Egress\_Active = FALSE
- 9. VERIFY Priority\_ Array = V4, ARRAY\_INDEX = PTY1
- 10. WRITE Present\_Value = NULL, PRIORITY = PTY2

-- Test for the Present\_Value at the specified property is changed

- 11. WRITE Present\_Value = V1, PRIORITY = PTY1
- 12. VERIFY Blink\_Warn\_Enable = TRUE
- 13. VERIFY Egress\_Time > 0

- 14. VERIFY Egress\_Active = FALSE
- 15. WRITE PROP REF = C1, PRIORITY = PTY1
- 16. BEFORE Internal Processing Fail Time CHECK (blink-warn occurred)
- 17. BEFORE Egress Active = FALSE
  - WRITE Present Value = V3, PRIORITY = PTY1
- 18. VERIFY Egress\_Active = FALSE
- 19. VERIFY Priority\_ Array = V3, ARRAY\_INDEX = PTY1

#### 7.3.1.X.9 Blink Warn WARN\_RELINQUISH Command Halted Test

Purpose: To verify blink-warn WARN RELINOUISH execution is halted when a higher priority entry is written or the Present Value at the specified priority is changed.

Test Concept: Select an object O1 that supports blink-warn commands. Configure O1 such that a blink-warn command will generate a blink-warn. Before the Egress timer expires, verify the specified actions clear the blinkwarn properties and the Priority\_Array is correctly changed.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL, slots numerically greater than PTY1 shall be V0 and no internal algorithms are issuing commands to O1 at any priority. Blink\_Warn\_Enable is TRUE, Egress\_Time is a non-zero value, Egress\_Active is FALSE and Relinquish Default is not V0.

	<b>Binary Lighting Output object</b>	Lighting Output object
PROP_REF	Present_Value	Present_Value or Lighting_Command
C1	WARN_RELINQUISH	-2.0 if PROP_REF = Present_Value, otherwise
		WARN_RELINQUISH
V0	OFF or NULL	0.0 or NULL
V1 to V3	ON	>1.0

Test Steps:

-- Test for a higher priority entry is written to a non NULL value

- 1. WRITE Present\_Value = V1, PRIORITY = PTY1
- 2. VERIFY Blink Warn Enable = TRUE
- 3. VERIFY Egress\_Time > 0
- 4. VERIFY Egress\_Active = FALSE
- WRITE PROP REF = C1, PRIORITY = PTY1 5.
- **BEFORE Internal Processing Fail Time** 6. CHECK (blink-warn occurred)
- 7. BEFORE Egress Active = FALSE WRITE Present Value = V2, PRIORITY = PTY2, a value not equal to 6 and less than PTY1
- VERIFY Egress\_Active = FALSE 8.
- 9. VERIFY Priority Array = NULL, ARRAY INDEX = PTY1
- 10. WRITE Present\_Value = NULL, PRIORITY = PTY2

-- Test for the Present\_Value at the specified property is changed

- 11. WRITE Present\_Value = V1, PRIORITY = PTY1
- VERIFY Blink\_Warn\_Enable = TRUE
   VERIFY Egress\_Time > 0
- 14. VERIFY Egress Active = FALSE
- 15. WRITE PROP REF = C1, PRIORITY = PTY1
- 16. BEFORE Internal Processing Fail Time CHECK (blink-warn occurred)
- 17. BEFORE Egress\_Active = FALSE WRITE Present\_Value = V3, PRIORITY = PTY1

- 18. VERIFY Egress\_Active = FALSE
- 19. VERIFY Priority\_ Array = V3, ARRAY\_INDEX = PTY1

[In BTL Specified Tests, add Binary Lighting Output object specific test 7.3.1.X41.10]

#### 7.3.2.X41.10 Binary Lighting Output Object Strike Count Tests

Dependencies: ReadProperty Service Execution Tests, 9.18; WriteProperty Service Execution Tests, 9.22.

Purpose: To verify that the properties of the Bianry Lighting Output object (O1) that tracks strike counts. Test Concept: The Present\_Value or Feedback\_Value of O1 can be used as the source S1 to increment Strike\_Count. S1 is transitioned from OFF to ON. The Strike\_Count property is checked to verify that it has been incremented. The Strike\_Count is reset and Time\_Of\_Strike\_Count\_Reset is checked to verify that it has been updated appropriately. Strike\_Count is set to a non-zero value and the Time\_Of\_Strike\_Count\_Reset is unchanged.

Configuration Requirements: O1 shall be configured such that the Present\_Value property is writable or another means of changing these properties shall be provided. Test Steps:

- 1 C1 = Strike\_Count
- 2. MAKE (S1 transition OFF to ON)
- 3. VERIFY (Strike Count = C1 + 1)
- 4. IF (Strike\_Count is writable) THEN MAKE (Strike\_Count = 0) VERUEX (Time\_Of\_Strike\_Count\_Reset = surrent less)
  - VERIFY (Time\_Of\_Strike\_Count\_Reset = current local time)
- IF (Strike\_Count is writable to a non-zero value) THEN MAKE (Strike\_Count > 0) VERIFY (Time\_Of\_Strike\_Count\_Reset is unchanged)

# **BTL-TP15.0-3.1.0 NM-CE-A Test Considerations**

Devices claiming support for the NM-CE-A BIBB must comply with the following section. This BIBB was revised in Protocol\_Revision 17.

#### **Overview:**

Addendum 135-2008v removed the NM-CE-A BIBB from all Device Profiles. This document makes needed changes in the BTL Test Package to claim NM-CE-A.

#### **Changes:**

[In BTL Checklist, add new Network Management - Connection Establishment - A]

Network Management - Connection Establishment - A		
	R	Base Requirements

[In BTL Test Plan, append to Section 10, Network Management]

### 10.X4 Network Management - Connection Establishment - A

### **10.X4.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

135.1	135.1-2013 - 10.5.3.1 - Establish-Connection-To-Network		
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	Must be executed.	
Test Directives			
	Testing Hints		
	Notes & Results		
135.1	-2013 - 10.5.3.2 - Discon	nect-Connection-To-Network	
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

# BTL-TP15.0-4.1.0 Read-only Recipient\_List Test Considerations

Device claiming a it has a read-only Recipient\_List property in a Notification class object must claim Protocol\_Revision 13 or higher and must comply with the following section.

[In BTL Checklist, in the Notification Class Object revise conformance code, and add indicated lineitem.]

Noti	Notification Class Object		
	R	Base Requirements	
	BTL-R	Supports DM-DDB-A	
	BTL-	Supports writable Recipient_List properties	
	$\mathbf{R}C^{I}$		
C <sup>1</sup> Supports read-only Recipient_List properties			
	<sup>1</sup> At least one of these options must be supported.		

[In BTL Test Plan, add a new section under Notification Class Object for Supports read-only Recipient\_List Properties. Entirely new sections proposed to be added in Test Plan use verbatim **bold**, or verbatim **bold**-*italic* throughout.]

### **3.17 Notification Class Object**

• • •

### 3.17.4 Supports read-only Recipient\_List Properties

The IUT supports read-only Recipient\_List properties.

BTL	BTL - 7.3.2.21.3.X9 Read-only Recipient_List for external Notification Forwarder objects		
	Test Method		
<b>Configuration</b> As per <b>BTL</b> Specified Tests.		As per BTL Specified Tests.	
	<b>Test Conditionality</b> Must be executed if the IUT does not claim support for Notification		
		Forwarder objects.	
	Test Directives		
	Testing Hints		
	Notes & Results		

In BTL Test Plan, modify existing Base Requirements section under Alarm and Event - Notification - Internal-B. Modified sections in Test Plan use yellow highlighted new material to preserve the verbatim **bold**, or verbatim **bold**. *italic*.

### **5.2 Alarm and Event - Notification - Internal-B**

### **5.2.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL - 7.3.1.10 - Event\_Enable Tests

-		
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If the IUT cannot be configured to meet the configuration requirements then this test shall be skipped.
	Test Directives	If Event Enrollment objects are supported, ensure this functionality is tested on Event Enrollment objects.
	Testing Hints	The BTL will apply this to a single object. The pretester should apply it to all objects that support alarm generation.
	Notes & Results	
BTL	- 7.3.1.12 - Notify_Type	Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If the IUT cannot be configured to meet the 135.1-2013 configuration requirements then this test shall be skipped.
	Test Directives	If Event Enrollment objects are supported, ensure this functionality is
		also tested on Event Enrollment objects.
	Testing Hints	
	Notes & Results	
135.1	-2009 - 8.4 - Confirmed	EventNotification Service Initiation Tests
	Test Method	
	Configuration	
	Test Conditionality	Must be executed unless IUT contains only read-only Recipient_List
		properties and does not claim Notification Forwarder objects.
		Any of the 8.4 tests can be used to ensure that the IUT properly
		generates Confirmed Eventivotification requests. The specific tests that
		can be executed are detailed under the test cases for the specific
		ConfirmedEventNetifications, then this test case shall be satisfied
	Test Directives	
	Test Directives	
	Notos & Dosulta	
	Notes & Results	
135 1		edEventNatification Service Initiation Tests
155.1	Test Method	
	Configuration	
	Test Conditionality	Must be executed
	Test Conditionality	Any of the 8.5 tests can be used to ensure that the IUT properly
		generates UnconfirmedEventNotification requests. The specific tests that
		can be executed are detailed under the test cases for the specific
		algorithms. As long as one of the tests is executed using
		UnconfirmedEventNotifications, then this test case shall be satisfied.
	Test Directives	
	Testing Hints	
	Notes & Results	

[In BTL Specified Tests, revise the Test Concepts for Recipient\_List tests, for special situations where Recipient\_List is read-only or static.]

### 7.3.2.21.3.1 ValidDays Test

Test Concept: The TD will select one instance of the Notification Class object and one instance of an event-generating object that is linked to it. The Recipient\_List of the Notification Class object shall contain a single recipient with the Valid Days parameter configured so that at least one day is TRUE and at least one day is FALSE. The properties of the event-generating object will be manipulated to cause the Event\_State to change from NORMAL to OFFNORMAL. The tester verifies that if the local date is one of the valid days a notification message is transmitted and the if local date is not a valid day then no notification message is transmitted. *For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects and are exclusively configured for all days (Valid Days set to all Days), this test shall be skipped. For devices that implement a writeable Recipient\_List property for all instances objects, and exclusively accept all days as the only permitted configuration, this test shall be skipped.* 

### 7.3.2.21.3.2 FromTime and ToTime Test

... Т...

. . .

Test Concept: The case where the local date and time fall within the window defined by the From Time and To Time parameters is covered by the ValidDays test in 7.3.2.21.3.1. This test uses the same IUT configuration and sets the local time to a value that is one of the ValidDays but outside of the window defined by the From Time and To Time parameters. The objective is to verify that an event notification message is not transmitted when the event is triggered. For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects and are exclusively configured for all times (From Time set to 00:00:00.0, To\_Time set to 23:59:59.90), this test shall be skipped. If all instances of writeable Notification Class Recipient\_List properties exclusively accept all times as the only permitted configuration, this test shall be skipped.

### 7.3.2.21.3.3 IssueConfirmedNotifications Test

...

Purpose: To verify that ConfirmedEventNotification messages are used if the Issue Confirmed Notifications parameter has the value TRUE and UnconfirmedEventNotification messages are used if the value is FALSE. If the IUT does not support both confirmed and unconfirmed event notifications this test may be *skipped* omitted. For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects, and there is a value of FALSE for the IssueConfirmedNotifications component in all instances, this test shall be skipped.

### 7.3.2.21.3.4 Transitions Test

Test Concept: The IUT is configured such that the Transitions parameter indicates that some event transitions are to trigger an event notification and some are not. Each event transition is triggered and the IUT is monitored to verify that notification messages are transmitted only for those transitions for which the Transitions parameter has a value of TRUE. For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects and are exclusively configured for all transitions (all bits in Transitions set to TRUE), this test shall be skipped. If all instances of writeable Notification Class Recipient\_List properties exclusively accept all transitions as the only permitted configuration, this test shall be skipped.

### 7.3.2.21.3.5 Recipient\_List Property Supports Device Identifier Recipients Test

Purpose: To verify that the Recipient\_List property of the Notification Class object supports the device form of the Recipient component and that the IUT is able to associate a MAC address with the Device Identifier. The intent is to ensure that the IUT is able to locate the specified alarm recipient and send notification to the specified recipient. This test shall be run if the IUT's Notification Class object's Recipient\_List property supports the BACnet object identifier form of BACnetRecipient.

Test Concept: The tester shall select a single event-generating object E in the IUT that references Notification Class object N. The tester shall add an entry into the Recipient\_List of the associated Notification Class object that specifies a Device Identifier, D, for a device that the IUT is not already aware of. The TD, acting as device D, shall be located on a different network than the IUT to ensure that the IUT is capable of binding to recipients located on any network. For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects and there is an address form of the Recipient component in all instances, this test shall be skipped.

Configuration Requirements: The TD shall be configured so that it does not execute WhoHas.

Test Steps:

- 1. WRITE N.RecipientList = ( {all days, all times, D, any process ID, FALSE, all transitions} )
- 2. MAKE (the event generating object, E, transition)
- 3. BEFORE Notification Fail Time plus the amount of time the IUT takes to perform device discovery RECEIVE UnconfirmedEventNotification-Request.

Process Identifier' -	(the valid process ID from stap 1)
Flocess Identifier -	(the valid process ID from step 1),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	Е,
'Time Stamp' =	(any valid time stamp),
'Notification Class' =	(N's instance),
'Priority' =	(any valid priority),
'Event Type' =	(any valid event type),
'Notify Type' =	ALARM   EVENT,
'AckRequired' =	TRUE   FALSE,
'From State' =	(any valid event state),
'To State' =	(any valid event state),
'Event Values' =	(values appropriate to the event type)

Notes to Tester: The IUT is expected to initiate one or more range-restricted WhoIs requests after the modification of the Recipient\_List but before the sending of the notification. The IUT might also need to perform other network discovery operations. Given that there are multiple approaches to the use of WhoIs for device discovery, the test only focuses on the IUT's ability to find device D and not on the specifics or timing of the WhoIs requests.

#### 7.3.2.21.3.6 Recipient\_List Property Supports Network Address Recipients

Purpose: To verify that the Recipient\_List property of the Notification Class object supports the address form of the Recipient component. The intent is to ensure that the IUT is able to send notifications to the specified recipient.

Test Concept: The tester shall select a single event-generating object E in the IUT that references Notification Class object N. The tester shall add an entry into the Recipient\_List of the associated Notification Class object that specifies a BACnetAddress A, where A is a unicast or is a local, remote, or global broadcast address. *For devices that implement a read-only Recipient\_List property for all instances of Notification Class objects and there is a Device Identifier form of the Recipient component in all instances, this test shall be skipped.* 

Test Steps:

- 1. WRITE N.RecipientList = ( {all days, all times, A, any process ID, FALSE, all transitions} )
- 2. MAKE (the event generating object, E, transition)
- 3. BEFORE Notification Fail Time RECEIVE UnconfirmedI

E UnconfirmedEventNotification-F	Request,
DESTINATION =	Ā,
'Process Identifier' =	(the valid process ID from step 1),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	Е,
'Time Stamp' =	(the current local time),
'Notification Class' =	(N's instance),
'Priority' =	(any valid priority),
'Event Type' =	(any valid event type),
'Notify Type' =	ALARM   EVENT,
'AckRequired' =	TRUE   FALSE,
'From State' =	(any valid event state),
'To State' =	(any valid event state),

'Event Values' =

[Add new test into BTL Specified Tests.]

### 7.3.2.21.3.X9 Read-only Recipient\_List for external Notification Forwarder Objects

Purpose: This test case verifies that a read-only Notification Class object Recipient\_List is configured with the content designed for external Notification Forwarder objects.

Test Concept: Read the Recipient\_List of the Notification Class objects and check that the length is 1, the Recipient is local broadcast, Valid Days are all days, From Time and To Time are the entire day, Process Identifier is 0, Issue Confirmed Notification is False and Transitions is set to all transitions. This test is only applied to IUT devices that have read-only Notification Class object Recipient\_List properties, and which do not contain internal Notification Forwarder objects.

Test Steps:

- 1. READ RL = Recipient\_List
- 2. VERIFY (RL is a list of length 1)
- 3. VERIFY (RL.Destination = { (1, 1, 1, 1, 1, 1, 1)--Valid Days

```
00:00:00.0 --From Time

23:59:59.99 --To Time

(BACnetAddress: network-number = 0, zero length mac-address)

0 --Process Identifier

False --Issue Confirmed Notifications

(True, True, True) --Transitions

})
```

# **BTL-TP15.0-5.1.0** Tests for the Lighting Output object

A device including a Lighting Output object must claim Protocol\_Revision 14 or higher and comply with the following section.

#### **Overview:**

Addendum 135-2010*i* added the Lighting Output object. This document makes needed changes in the BTL Test Package to claim Lighting Output object.

These changes are not contained in any SSPC proposal.

[In BTL Checklist, add Lighting Output object type to Section 3, Objects]

Support	Listing	Option
Ligl	hting Outpu	ıt Object
	R	Base Requirements
	R	Supports command prioritization
	R	Supports all BACnetLightingOperations
	S	Supports writable Out_Of_Service properties

Support	Listing	Option
	0	Supports blink-warn
	0	Supports Transition property
	0	Supports Feedback_Value property
	0	Supports Min_Actual_Value and Max_Actual_Value properties
	0	Contains an object with Reliability_Evaluation_Inhibit Property

[In BTL Test Plan, add Lighting Output object tests in section 3.X54. In the following addition of new clauses to the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold**-*italic* as shown.]

### **3.X54 Lighting Output Object**

### **3.X54.1 Base Requirements**

Base requirements must be met by any IUT that can contain Lighting Output objects.

BTL - 7.3.2.X54.21 - Lighting Output Tracking Test			
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL - 7.3.2.X54.22 - Lighting Output Present Value between 0.0 and 1.0 Test			
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 3.X54.2 Supports Command Prioritization

135.1-2013 - 7.3.1.2 - Relinquish Default Test			
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	If no object can be made to meet the configuration requirements, this test	
		shall be skipped.	
	Test Directives		
	Testing Hints		
	Notes & Results		
-------	-------------------------------	---------------------------	
125.1	2012 7212 Comm	and Drivritination Test	
135.1	<u>-2013 - 7.3.1.3 - Comm</u>	and Prioritization Test	
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	Must be executed.	
	<b>Test Directives</b>		
	Testing Hints		
	Notes & Results		

## 3.X54.3 Supports all BACnetLightingOperations

BTL	L -7.3.2.X54.31 Lighting Command Operation NONE Test		
	Test Method	Manual	
	Configuration	As per <b>BTL Specified Tests</b> .	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL ·	- 7.3.2.X54.32 Lighting	Command Operation FADE_TO Test	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives	Repeat the test by using the BACnetLightingCommand without the	
		optional fields (priority and fade-time) and check that PTY1=	
		Lighting_Command_Default_Priority and fade-time = Default_Fade_Time	
	Testing Hints		
	Notes & Results		
DIDIT	7 2 2 V54 22 Linhting		
BIL	- 7.3.2.A54.35 Lighting	Command Operation RAMP_TO Test	
BIL	Test Method	Manual	
BIL	Test Method Configuration	Command Operation RAMP_10 Test         Manual         As per BTL Specified Tests.	
BIL	Test Method Configuration Test Conditionality	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.	
BIL	Test Method Configuration Test Conditionality Test Directives	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the	
BIL	Test Method Configuration Test Conditionality Test Directives	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=	
BIL	Test Method Configuration Test Conditionality Test Directives	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_	
BIL	Test Method Configuration Test Conditionality Test Directives	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BIL	Test Method Configuration Test Conditionality Test Directives	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BIL	7.5.2.X34.55 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BIL	7.5.2.X34.55 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BIL	7.5.2.X34.55 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting	Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual         As per BTL Specified Tests.	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration         Test Method         Configuration         Test Conditionality	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual         As per BTL Specified Tests.         Must be executed.	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration         Test Conditionality         Test Method         Configuration         Test Conditionality         Test Directives	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the external field (criming and property) and check that DTY1	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration         Test Conditionality         Test Method         Configuration         Test Conditionality         Test Directives	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and check that PTY1=	
BTL	7.3.2.X54.35 Lighting         Test Method         Configuration         Test Conditionality         Test Directives         Testing Hints         Notes & Results         - 7.3.2.X54.34 Lighting         Test Method         Configuration         Test Conditionality         Test Method         Configuration         Test Conditionality         Test Directives	Command Operation RAMP_TO Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and ramp-rate = Default_         Ramp_Rate         Command Operation STEP_UP Test         Manual         As per BTL Specified Tests.         Must be executed.         Repeat the test by using the BACnetLightingCommand without the optional fields (priority and ramp-rate) and check that PTY1=         Lighting_Command_Default_Priority and step-increment = Default_	

	Testing Hints	
	Notes & Results	
BTL ·	- 7.3.2.X54.35 Lighting	Command Operation STEP_DOWN Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	Must be executed.
	Test Directives	Repeat the test by using the BACnetLightingCommand without the
		optional fields (priority and ramp-rate) and check that PTY1=
		Lighting_Command_Default_Priority and step-increment = Default_
		Step_increment
	Testing Hints	
	Notes & Results	
BTL ·	- 7.3.2.X54.36 Lighting	Command Operation STEP_ON Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	Repeat the test by using the BACnetLightingCommand without the
		optional fields (priority and ramp-rate) and check that PTY1=
		Lighting_Command_Default_Priority and step-increment = Default_ step-
		increment
	Testing Hints	
	Notes & Results	
BTL ·	- 7.3.2.X54.37 Lighting	Command Operation STEP_OFF Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	Repeat the test by using the BACnetLightingCommand without the
		optional fields (priority and ramp-rate) and check that PTY1=
		Lighting_Command_Default_Priority and step-increment = Default_ step-
		increment
	Testing Hints	
	Notes & Results	

## 3.X54.4 Supports Writable Out\_Of\_Service Properties

The Out\_Of\_Service property in Lighting Output objects contained in the IUT are writable.

135.1	135.1-2013 - 7.3.1.1 - Out_Of_Service, Status_Flags, and Reliability Tests		
Test Method Manual		Manual	
ſ	Configuration		
ſ	Test Conditionality Must be executed.		
ſ	Test Directives         The test shall be executed using an Lighting Output object		
Testing Hints			
I	Notes & Results		

## 3.X54.5 Supports blink-warn

BTL ·	- 7.3.1.X.1 - Blink War	n WARN Command Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	Must be executed.
	Test Directives	Must be executed using both the Present Value and Lighting Command
		commands.
	Testing Hints	
	Notes & Results	
BTL	- 7.3.1.X.2 - Blink War	n WARN OFF Command Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests
	Test Conditionality	Must be executed
	Test Directives	Must be executed using both the Present Value and Lighting Command
	Test Directives	commands
		communds.
	Testing Hints	
	Notos & Rosults	
	Tioles & Results	
BTI	- 7 3 1 X 3 - Blink War	n WARN RELINCIUSH Command Test
DIL	Test Method	
	Configuration	Manual
	Test Conditionality	As per <b>DIL</b> specified Tesis.
	Test Conditionality	Must be executed using both the Present Volue and Lighting Command
	Test Directives	Must be executed using both the Present_value and Lighting_Command
		commands.
	Testing Hints	
	Noton & Domite	
	Notes & Results	
DTI	721VA Dlink Wor	n STOD Commond Tost
DIL	- 7.3.1.A.4 - DHIIK WAL	Manual
	Configuration	Manual
	Tost Conditionality	As per <b>DIL Specifica Tesis</b> . Must be executed
	Test Conditionality	IVIUSI DE EXECUTEU.
	Test Directives	Repeat the test with wAKIN_OFF and WAKIN_KELINQUISH commands
	Noton & Day 14	
	notes & Results	
DTI	721V5 Disal-War	n WADN Command Failure Test
DIL	- 1.3.1.A.3 - BIINK WAR	n waann command fanure lest
	Configure	Ivianual
	Configuration	As per <b>DIL Specifica Tesis</b> .
	Test Conditionality	Must de executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

The Blink\_Warn\_Enable property in Lighting Output is writable or can be changed to TRUE by other means.

BTL	7.3.1.X.6 - Blink Warn WARN_OFF Command Failure Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 7.3.1.X.7 - Blink War	n WARN_RELINQUISH Command Failure Test	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 7 3 1 X 8 - Blink War	n WARN OFF Command Halted Test	
DIL	Test Method	Manual	
	Configuration	As per <b>BTL Specified Tests</b> .	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	IL - 7.3.1.X.9 - Blink Warn WARN_RELINQUISH Command Halted Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

## 3.X54.6 Supports Transition property

The supports transition section and tests for: RAMP transition executes a ramp operation from the Tracking\_Value to the target level using the ramp rate specified in Default\_Ramp\_Rate.

BTL	3TL - 7.3.2.X54.41 Transition None Test		
	Test Method Manual		
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	<b>Test Directives</b>		
	Testing Hints		
	Notes & Results		
BTL ·	BTL - 7.3.2.X54.42 Transition Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	

	Test Directives		
	Testing Hints		
	Notes & Results		

## **3.X54.7** Supports Feedback\_Value property

The IUT contains Lighting Output Objects in which the Feedback\_Value property is supported.

BTL	TL - 7.3.2.X54.51 - Feedback_Value Clamping Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

## 3.X54.8 Supports Min\_Actual\_Value and Max\_Actual\_Value properties

The IUT contains Lighting Output Objects in which the the Min\_Actual\_Value and Max\_Actual\_Value properties are supported.

BIL	BTL - 7.3.2.X54.61 - Min_Actual_Value and Max_Actual_Value Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	BTL - 7.3.2.X54.62 - Min_Actual_Value and Max_Actual_Value ScalingTest		
	-	0	
	Test Method	Manual	
	Test Method Configuration	Manual As per <i>BTL Specified Tests</i> .	
	Test Method Configuration Test Conditionality	Manual       As per BTL Specified Tests.       Must be executed.	
	Test Method Configuration Test Conditionality Test Directives	Manual       As per BTL Specified Tests.       Must be executed.	
	Test Method Configuration Test Conditionality Test Directives Testing Hints	Manual       As per BTL Specified Tests.       Must be executed.	
	Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	Manual       As per BTL Specified Tests.       Must be executed.	
	Test MethodConfigurationTest ConditionalityTest DirectivesTesting HintsNotes & Results	Manual       As per BTL Specified Tests.       Must be executed.	

## 3.X54.9 Contains an object with Reliability\_Evaluation\_Inhibit Property

The IUT contains, or can be made to contain, a Reliability\_Evaluation\_Inhibit property that is configurable to a value of TRUE.

BTL - 7.3.1.X8.1 - Reliability_Evaluation_Inhibit Test		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.

	Test Conditionality	If no object exists in the IUT for which fault conditions can be generated
		then this test shall be skipped.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.1.X8.2 - Reliability	_Evaluation_Inhibit Summarization Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If no object exists in the IUT for which fault conditions can be generated
		then this test shall be skipped.
	Test Directives	
	Testing Hints	
	Notes & Results	

#### Changes:

[In BTL Test Plan, add "Supports COV for Lighting Output Objects " tests" in section 4.10.X. In the following modification of clauses of the Test Plan, further changes in a test name, which had already been earlier changed for the version in BTL Specified Tests (in *bold-italic* as shown), are indicated in *red-italic*, with the rest of these new clauses of the Test Plan in plain text, verbatim **bold**, or verbatim *bold-italic* as shown.]

## 4.10.X54 Supports COV for Lighting Output Objects

The IUT supports change of value notifications for at least one object of type Lighting Output.

BTL - 8.2.1 - Change of Value Notification from an Analog Input, Analog Output, <i>Lighting</i>			
Output, and Analog Value, Large Analog Value, Integer Value, and Positive Integer Value Object			
Prese	nt_Value Property		
	Test Method Manual		
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints	This may be skipped if 8.3.1 is executed against a Lighting Output object.	
	Notes & Results		
BTL - 8.2.2 - Change of Value Notification from an Analog Input, Analog Output, <i>Lighting Output</i> , and Analog Value, <i>Large Analog Value</i> , <i>Integer Value</i> , and <i>Positive Integer Value</i> Object Status Flags Property			
	Test Method Manual		
	Configuration         As per BTL Specified Tests. The selected object must be a Light Output.		
	Test Conditionality Must be executed.		
	Test Directives		
Testing Hints         This may be skipped if 8.3.2 is executed against a Lighting Or object.		This may be skipped if 8.3.2 is executed against a Lighting Output object.	
	Notes & Results		

BTL - 8.3.1 - Change of Value Notification from an Analog Input, Analog Output, *Lighting Output*, and-Analog Value, *Large Analog Value, Integer Value, and Positive Integer Value* Object Present\_Value Property

I I USU	m_vanue i roperty			
	Test Method	Manual		
	Configuration	As per BTL Specified Tests. The selected object must be a Lighting		
	-	Output.		
	Test Conditionality	Must be executed.		
	Test Directives			
	Testing Hints	This may be skipped if 8.2.1 is executed against a Lighting Output object.		
	Notes & Results			
<b>BTL - 8.3.2 - Change of Value Notification from an Analog Input, Analog Output, </b> <i>Lighting Output,</i> and Analog Value, <i>Large Analog Value, Integer Value, and Positive Integer Value</i> Object Status Flags Property				
	Test Method	Manual		
	Configuration         As per BTL Specified Tests. The selected object must be a Light           Output.         Output.			
	Test Conditionality	Must be executed.		
	Test Directives			
	Testing Hints	This may be skipped if 8.2.2 is executed against a Lighting Output object.		
	Notes & Results			

#### **Changes:**

[In BTL Specified Tests, add new Lighting Output object specific tests in section 7.3.2.X54]

#### 7.3.2.X54.21 - Lighting Output Tracking Test

Purpose: To verify that the Tracking\_Value property follows the Present\_Value property.

Test Concept: Write to the Present\_Value of a Lighting Output object, O1, and verify that the Tracking\_Value property follows Present\_Value once In-Progress returns to IDLE.

Configuration Requirements: The IUT shall be configured with a lighting output O1 that can be observed during the test. O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1 and Out\_Of\_Service = FALSE. Any scaling information that may be needed to verify that the value is reasonable shall also be provided.

Test Steps:

- 1. WRITE Present\_Value = 100, PRIORITY = PTY1
- 2. VERIFY Present\_Value = 100
- 3. WHILE (In\_Progress <> IDLE) DO {
- 4. VERIFY Tracking Value = 100
- 5. WRITE Present Value = 1, PRIORITY = PTY1
- 6. VERIFY Present\_Value = 1
- 7. WHILE (In\_Progress <> IDLE) DO {
  }

- 8. VERIFY Tracking\_Value = 1
- 9. WRITE Present\_Value = 0, PRIORITY = PTY1
- 10. VERIFY Present\_Value = 0
- 11. WHILE (In\_Progress <> IDLE) DO {
- 12. VERIFY Tracking Value = 0

#### 7.3.2.X54.22 - Lighting Output Present Value between 0.0 and 1.0 Test

Purpose: To verify that writing a value numerically greater than 0.0 but less than 1.0 to Present\_Value shall result in Present\_Value taking on the value 1.0.

Test Concept: Select a value, V1, which is numerically greater than 0.0 and less than 1.0. Write V1 to Present\_Value and verify that Present\_Value takes on the value 1.0.

Configuration Requirements: The Lighting Output object, O1, shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. Present\_Value shall be different from 1.0.

Test Steps:

- 1. VERIFY Present\_Value <> 1.0
- 2. WRITE Present\_Value = a value numerically greater than 0.0 but less than 1.0
- 3. VERIFY Present\_Value = 1.0

#### 7.3.2.X54.31 Lighting Command Operation NONE Test

Purpose: To verify that the IUT can execute WriteProperty service requests when an attempt is made to write a value that is outside of the supported range.

Test Concept: The TD writes the Lighting Command Operation NONE to the IUT, and expects Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE

Test Steps:

- 1. VERIFY (Object1), P1 = (the value defined for this property in the EPICS),
- TRANSMIT WriteProperty-Request, 'Object Identifier' = 01 'Property Identifier' = Lighting\_Command
- 'Property Value' = NONE 3. RECEIVE BACnet-Error PDU.
  - Error Class = PROPERTY.
  - Error Code = VALUE\_OUT\_OF\_RANGE
- 4. VERIFY (Object1), Lighting\_Command = (the value defined for this property in the EPICS)

#### 7.3.2.X54.32 Lighting Command Operation FADE\_TO Test

Purpose: To verify the correct operation of FADE\_TO lighting command by observing the value of Present\_Value, In\_Progress and Tracking\_Value.

Test Concept: The TD writes to the Present\_Value at each end of the range (i.e. 0% or 100%), and then writes to the Lighting Command Operation with FADE\_TO with a long enough fade-time to allow In\_Progress and Tracking\_Value to be observed while set to FADE\_ACTIVE. The Tracking\_Value will be checked at the end of the fade to verify that it tracked the target level. The IUT shall be tested for fade up (0% to 100%) and fade down (100% to 0%).

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. V1 > 1 and V2 < 100%

Test Steps:

-- Start with 0% Present\_Value to test fade up

- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0

-- Write a FADE\_TO command (operation, target-level, priority, fade-time)

- 5. WRITE Lighting\_Command = (FADE\_TO, V1, PTY1, FT)
- 6. WAIT Internal Processing Fail Time
- 7. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value = V1

-- In a half way of fading up, check In\_Progress and Tracking\_Value

- 9. WAIT FT/2
- 10. VERIFY In\_Progress = FADE\_ACTIVE,
- 11. VERIFY Tracking\_Value ~=V1 / 2
- 12. WAIT FT/2

-- When fading up is completed, check In\_Progress and Tracking\_Value

- 13. VERIFY In\_Progress = IDLE
- 14. VERIFY Tracking\_Value = V1

-- Now repeat the test with 100% Present\_Value to test fade down

- 15. WRITE Present\_Value = 100, ARRAY\_INDEX = PTY1
- 16. VERIFY Present\_Value = 100
- 17. WAIT Internal Processing Fail Time
- 18. VERIFY Tracking\_Value = 100

-- Write a FADE\_TO command (operation, target-level, priority, fade-time)

- 19. WRITE Lighting\_Command = (FADE\_TO, V2, PTY1, FT)
- 20. WAIT Internal Processing Fail Time
- 21. VERIFY Priority\_ Array = V2, ARRAY\_INDEX = PTY1
- 22. VERIFY Present\_Value = V2

-- In a half way of fading down, check In\_Progress and Tracking\_Value

- 23. WAIT FT/2
- 24. VERIFY In\_Progress = FADE\_ACTIVE,
- 25. VERIFY Tracking\_Value ~= V1 / 2
- 26. WAIT FT/2

-- When fading down is completed, check In\_Progress and Tracking\_Value

- 27. VERIFY In\_Progress = IDLE
- 28. VERIFY Tracking\_Value = V2

#### 7.3.2.X54.33 Lighting Command Operation RAMP\_TO Test

Purpose: To verify the correct operation of RAMP\_TO lighting command by observing the value of Present\_Value, In\_Progress and Tracking\_Value.

Test Concept: The TD writes to Present\_Value at each end of the range (i.e. 0% or 100%), and then writes to the Lighting Command Operation with RAMP\_TO with a slow enough ramp rate to allow In\_Progress and Tracking\_Value to be observed while set to RAMP\_ACTIVE. The Tracking\_Value will be checked at the end of the ramp to verify that it tracked the target level. The IUT shall be tested for ramp up (0% to 100%) and ramp down (100% to 0%).

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. V1 > 1 and V2 < 100%

Test Steps:

- -- Start with 0% Present\_Value to test ramp up
- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0
- -- Write a RAMP\_TO command (operation, target-value, priority, ramp-rate)
- 5. WRITE Lighting\_Command = (RAMP\_TO, V1, PTY1, any valid rate)
- 6. WAIT Internal Processing Fail Time
- 7. VERIFY Priority\_ Array = V1, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value =V1

-- Check In\_Progress while ramping up

9. VERIFY In\_Progress = RAMP\_ACTIVE

-- Make sure that Tracking\_Value increases with the ramp-rate

- 10. WHILE (In\_Progress <> IDLE) DO {
- 11. VERIFY Tracking\_Value > 0 < V1
- 12. CHECK (Tracking\_Value is increasing with the ramp-rate)}

-- When ramping up is completed, check In\_Progress and Tracking\_Value

- 13. VERIFY In\_Progress = IDLE
- 14. VERIFY Tracking\_Value = V1

-- Now repeat the test with 100% Present\_Value to test ramp down

- 15. WRITE Present\_Value = 100, ARRAY\_INDEX = PTY1
- 16. VERIFY Present\_Value = 100
- 17. WAIT Internal Processing Fail Time
- 18. VERIFY Tracking\_Value = 100

-- Write a RAMP\_TO command (operation, target-value, priority, ramp-rate)

- 19. WRITE Lighting\_Command = (RAMP\_TO, V2, PTY1, any valid rate)
- 20. WAIT Internal Processing Fail Time
- 21. VERIFY Priority\_ Array = V2, ARRAY\_INDEX = PTY1
- 22. VERIFY Present\_Value = V2

-- Check In\_Progress while ramping up 23. VERIFY In\_Progress = RAMP\_ACTIVE,

-- Make sure that Tracking\_Value decreases with the ramp-rate

- 24. WHILE (In\_Progress <> RAMP\_ACTIVE) DO {
- 25. VERIFY Tracking\_Value < 0 > V2
- 26. CHECK (Tracking\_Value is decreasing with the ramp-rate)}

-- Check In\_Progress and Tracking\_Value

- 27. VERIFY In\_Progress = IDLE
- 28. VERIFY Tracking\_Value = V2

#### 7.3.2.X54.34 Lighting Command Operation STEP\_UP Test

Purpose: To verify the correct operation of STEP\_UP lighting command by observing the value of Present\_Value, In\_Progress and Tracking\_Value.

Test Concept: The TD writes to Present\_Value at 0%, and then writes to the Lighting Command Operation with STEP\_UP and any step increment. The Tracking\_Value shall remain at 0% to ignore the operation. Next, the TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment greater than 99%, the Tracking\_Value shall be 100%. The TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment greater than 99%, the Tracking\_Value shall be 100%. The TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment less than 99%, the Tracking\_Value shall be 1% plus the step increment.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1.

Test Steps:

-- Start with 0% Present\_Value

- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0

-- Write a STEP\_UP command (operation, priority, step-increment)

- 5. WRITE Lighting\_Command = (STEP\_UP, PTY1, any valid value)
- 6. WAIT Internal Processing Fail Time

-- Confirm that the command was ignored since Tracking\_Value was 0

- 7. VERIFY Priority\_Array = 0, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value = 0
- 9. VERIFY Tracking\_Value = 0

-- Now test with Tracking\_Value >0

- 10. WRITE Present\_Value = 1, ARRAY\_INDEX = PTY1
- 11. VERIFY Present\_Value = 1
- 12. WAIT Internal Processing Fail Time
- 13. VERIFY Tracking\_Value = 1
- -- Keep stepping up while continuously checking Priority\_Array, Present\_Value and Tracking\_Value
- 14. REPEAT  $X = (1 \text{ through } (100 \text{ step-increment}) \text{ by step-increment}) \text{ DO} \{$

WRITE Lighting\_Command = (STEP\_UP, PTY1, any valid value) WAIT **Internal Processing Fail Time** VERIFY Priority\_ Array = X + step-increment, ARRAY\_INDEX = PTY1 VERIFY Present\_Value = X + step-increment VERIFY Tracking\_Value = X + step-increment

-- Now step up one more time to confirm that the values will not exceed 100

15. WRITE Lighting\_Command = (STEP\_UP, PTY1, any valid value)

- 16. WAIT Internal Processing Fail Time
- 17. VERIFY Priority\_ Array = 100, ARRAY\_INDEX = PTY1
- 18. VERIFY Present\_Value =100
- 19. VERIFY Tracking\_Value = 100

#### 7.3.2.X54.35 Lighting Command Operation STEP\_ DOWN Test

Purpose: To verify that writing this Lighting Command Operation is reflected in the Tracking\_Value, that writes resulting in a step below 1% are limited to 1%, and that this command is ignored if the Tracking\_Value is 0.0%.

Test Concept: The TD writes to Present\_Value at 0%, and then writes to the Lighting Command Operation with STEP\_DOWN and any step increment. The Tracking\_Value shall remain at 0%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command Operation with STEP\_DOWN and a step increment greater than 99%, the Tracking\_Value shall be 1%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command operation with STEP\_DOWN and a step increment greater than 99%, the Tracking\_Value shall be 1%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command Operation with STEP\_DOWN and a step increment less than 99%, the Tracking\_Value shall be 100% minus the step increment.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1.

Test Steps:

-- Start with 0% Present\_Value

- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0

-- Write a STEP\_DOWN command (operation, priority, step-increment)

- 5. WRITE Lighting\_Command = (STEP\_ DOWN, PTY1, any valid value)
- 6. WAIT Internal Processing Fail Time

-- Confirm that the command was ignored since Tracking\_Value was 0

- 7. VERIFY Priority\_ Array = 0, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value = 0
- 9. VERIFY Tracking\_Value = 0

-- Now test with Tracking\_Value = 100

- 10. WRITE Present\_Value = 100, ARRAY\_INDEX = PTY1
- 11. VERIFY Present\_Value = 100
- 12. WAIT Internal Processing Fail Time
- 13. VERIFY Tracking\_Value =100

-- Keep stepping down while continuously checking Priority\_Array, Present\_Value and Tracking\_Value 14. REPEAT X = (100 through (1 + step-increment) by step-increment) DO{

WRITE Lighting\_Command = (STEP\_ DOWN, PTY1, any valid value) WAIT **Internal Processing Fail Time** VERIFY Priority\_ Array = X - step-increment, ARRAY\_INDEX = PTY1 VERIFY Present\_Value = X - step-increment VERIFY Tracking\_Value = X - step-increment

-- Now step down one more time to confirm that the values will not go down below 1

- 15. WRITE Lighting\_Command = (STEP\_ DOWN, PTY1, any valid value)
- 16. WAIT Internal Processing Fail Time
- 17. VERIFY Priority\_ Array = 1, ARRAY\_INDEX = PTY1
- 18. VERIFY Present\_Value =1
- 19. VERIFY Tracking\_Value = 1

#### 7.3.2.X54.36 Lighting Command Operation STEP\_ON Test

Purpose: To verify that writing this Lighting Command Operation is reflected in the Tracking\_Value, that this command will set the Tracking\_Value to 1% if the Tracking\_Value is 0.0%, and that it otherwise adheres to STEP\_UP.

Test Concept: The TD writes to Present\_Value at 0%, and then writes to the Lighting Command Operation with STEP\_UP and any step increment. The Tracking\_Value shall be 1%. The TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment greater than 99%, the Tracking\_Value shall be 100%. The TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment greater than 99%, the Tracking\_Value shall be 100%. The TD writes to Present\_Value at 1%, and then writes to the Lighting Command Operation with STEP\_UP and a step increment less than 99%, the Tracking\_Value shall be 1% plus the step increment.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1.

Test Steps:

-- Start with 0% Present\_Value

- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0

-- Write a STEP\_ON command (operation, priority, step-increment)

- 5. WRITE Lighting\_Command = (STEP\_ON, PTY1, any valid values)
- 6. WAIT Internal Processing Fail Time

-- Confirm that the Present\_Value and Tracking\_Value became 1

- 7. VERIFY Priority\_ Array = 1, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value = 1
- 9. VERIFY Tracking\_Value = 1

-- Keep stepping on while continuously checking Priority\_Array, Present\_Value and Tracking\_Value

10. REPEAT X = (1 through (100 - step-increment))

WRITE Lighting\_Command = (STEP\_ON, PTY1, any valid values) WAIT **Internal Processing Fail Time** VERIFY Priority\_Array = X + step-increment, ARRAY\_INDEX = PTY1 VERIFY Present\_Value = X + step-increment VERIFY Tracking Value = X + step-increment

-- Now step on one more time to confirm that the values will not exceed 100

- 11. WRITE Lighting\_Command = (STEP\_ON, PTY1, any valid values)
- 12. WAIT Internal Processing Fail Time
- 13. VERIFY Priority\_ Array = 100, ARRAY\_INDEX = PTY1
- 14. VERIFY Present\_Value =100
- 15. VERIFY Tracking\_Value = 100

#### 7.3.2.X54.37 Lighting Command Operation STEP\_ OFF Test

Purpose: To verify that writing this Lighting Command Operation is reflected in the Tracking\_Value, that writes resulting in a step below 1% are limited to 1%, and that this command is ignored if the Tracking\_Value is 0.0%.

Test Concept: The TD writes to Present\_Value at 0%, and then writes to the Lighting Command Operation with STEP\_DOWN and any step increment. The Tracking\_Value shall remain at 0%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command Operation with STEP\_DOWN and a step increment greater than 99%, the Tracking\_Value shall be 1%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command operation with STEP\_DOWN and a step increment greater than 99%, the Tracking\_Value shall be 1%. The TD writes to Present\_Value at 100%, and then writes to the Lighting Command Operation with STEP\_DOWN and a step increment less than 99%, the Tracking\_Value shall be 100% minus the step increment.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1.

Test Steps:

- -- Start with 0% Present\_Value
- 1. WRITE Present\_Value = 0, ARRAY\_INDEX = PTY1
- 2. VERIFY Present\_Value = 0
- 3. WAIT Internal Processing Fail Time
- 4. VERIFY Tracking\_Value = 0

-- Write a STEP\_OFF command (operation, priority, step-increment)

5. WRITE Lighting\_Command = (STEP\_OFF, PTY1, step-increment)

6. WAIT Internal Processing Fail Time

-- Confirm that the command was ignored since Tracking\_Value was 0

- 7. VERIFY Priority\_ Array = 0, ARRAY\_INDEX = PTY1
- 8. VERIFY Present\_Value = 0
- 9. VERIFY Tracking\_Value = 0

-- Now test with Tracking\_Value = 100

- 10. WRITE Present\_Value = 100, ARRAY\_INDEX = PTY1
- 11. VERIFY Present\_Value = 100
- 12. WAIT Internal Processing Fail Time
- 13. VERIFY Tracking\_Value =100

-- Keep stepping off while continuously checking Priority\_Array, Present\_Value and Tracking\_Value

14. REPEAT X = (100 through (1 + step-increment))

WRITE Lighting\_Command = (STEP\_ OFF, PTY1, step-increment) WAIT **Internal Processing Fail Time** VERIFY Priority\_ Array = X - step-increment, ARRAY\_INDEX = PTY1 VERIFY Present\_Value = X - step-increment VERIFY Tracking Value = X - step-increment

-- Confirm that the Present\_Value and Tracking\_Value become 0 when STEP OFF command is executed while Tracking\_Value is 1

15. WRITE Lighting\_Command = (STEP\_ OFF, PTY1, step-increment)

- 16. WAIT Internal Processing Fail Time
- 17. VERIFY Priority\_ Array = 0, ARRAY\_INDEX = PTY1
- 18. VERIFY Present\_Value = 0
- 19. VERIFY Tracking\_Value = 0

#### 7.3.2.X54.41 Transition None test

Purpose: To verify that the Tracking\_Value property immediately follows the Present\_Value property if Transition is NONE.

Test Concept: Setup a Lighting Output object, O1, to use its complete supported value range. Set Present\_Value to the highest supported value, and then to the lowest supported value, verifying that there is no delay in the transitions.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. If present, Min\_Actual\_Value shall be set to 1, and Max\_Actual\_Value shall be set to 100. Transition shall be set to NONE.

Test Steps:

- 1. VERIFY Transition = NONE
- 2. VERIFY In\_Progress = IDLE
- 3. WRITE Present\_Value = 100, ARRAY\_INDEX = PTY1
- 4. VERIFY In\_Progress = IDLE
- 5. VERIFY Tracking\_Value = 100
- 6. WRITE Present\_Value = 1, ARRAY\_INDEX = PTY1
- 7. VERIFY In\_Progress = IDLE
- 8. VERIFY Tracking\_Value = 1

#### 7.3.2.X54.42 Transition Test

Purpose: To verify that the Lighting Output object transitions using the configured function and transitions at the configured speed when Transition is set to either FADE or RAMP.

Test Concept: Setup a Lighting Output object, O1, to use fading or ramping as the default transition method. Present\_Value is changed to V1 which is larger than the initial Present\_Value, V0, so that the output will fade or ramp up. Halfway through the process, verify that Tracking\_Value is approximately equal to the value halfway between V0 and V1. The physical output shall also be verified that it is fading or ramping from V0 to V1. When the process completes, verify that Tracking\_Value reached V1. Repeat the process fading or ramping down from V1 to V2.

Configuration Requirements: O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. The Transition property is set to FADE or RAMP, Present\_Value is V0 and In\_Progress is IDLE.

To test FADE functionality, T is FADE, A is FADE\_ACTIVE, W1 and W2 are (Default\_Fade\_Time / 2), and Default\_Fade\_Time is sufficiently large so as to allow the intermediate progress checks.

To Test RAMP functionality, T is RAMP, A is RAMP\_ACTIVE, W1 is (( $(V1 - V0) / Default_Ramp_Rate) / 2$ ), W2 is (( $(V1 - V2) / Default_Ramp_Rate) / 2$ ), and Default\_Ramp\_Rate is sufficiently small so as to allow the intermediate progress checks.

Test Steps:

- 1. VERIFY Transition = T
- 2. VERIFY In\_Progress = IDLE
- 3. V0 = READ Present\_Value
- 4. WRITE Present\_Value = V1, ARRAY\_INDEX = PTY1
- 5. VERIFY Present\_Value = V1
- 6. WAIT W1
- 7. VERIFY Tracking\_Value  $\sim = (V1 + V0) / 2$
- 8. VERIFY In\_Progress = A
- 9. CHECK (the physical output is fading from V0 to V1)
- 10. WAIT W1
- 11. VERIFY In\_Progress = IDLE
- 12. VERIFY Tracking\_Value = V1
- 13. WRITE Present\_Value = V2, ARRAY\_INDEX = PTY1
- 14. VERIFY Present\_Value = V2
- 15. WAIT W2
- 16. VERIFY Tracking\_Value  $\sim = (V2 + V1) / 2$
- 17. VERIFY In\_Progress = A
- 18. CHECK (the physical output is fading V1 to V2)
- 19. WAIT W2
- 20. VERIFY In\_Progress = IDLE
- 21. VERIFY Tracking\_Value = V2

```
7.3.2.X54.51 - Feedback_Value Clamping Test
```

Purpose: To verify that the Feedback\_Value remains in the normalized range when the physical lighting output is outside the normalized range.

Test Concept: Set the normalized range to be the largest range supported by the device. Make the physical output be above the normalized range by setting it to the maximum supported value and then shrinking the normalized range. The Feedback\_Value is immediately tested to verify that it takes on the value 100.

Reset the normalized range. Make the physical output be below the normalized range by setting it to the minimum supported value and then shrinking the normalized range. The Feedback\_Value is immediately tested to verify that it takes on the value 1.

Configuration Requirements: The Lighting Output object, O1, shall be configured to transition slowly when Present\_Value changes, such as by ramping, fading or stepping, if possible.

O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1.

Test Steps:

-- Verify Feedback\_Value when output is above Max\_Actual\_Value

- 1. WRITE Max\_Actual\_Value = 100
- 2. WRITE Min\_Actual\_Value = 1
- 3. WRITE Present\_Value = 100, PRIORITY = PTY1
- 4. WHILE In\_Progress <> IDLE { }
- 5. WRITE Max\_Actual\_Value = (Lowest supported Max\_Actual\_Value)
- 6. VERIFY Feedback\_Value = 100

-- Verify Feedback\_Value when output is below Min\_Actual\_Value

- 7. WRITE Max\_Actual\_Value = 100
- 8. WRITE Min\_Actual\_Value = 1
- 9. WRITE Present\_Value = 1, PRIORITY = PTY1
- 10. WHILE In\_Progress <> IDLE { }
- 11. WRITE Min\_Actual\_Value = (Highest supported Min\_Actual\_Value)
- 12. VERIFY Feedback\_Value = 1

#### 7.3.2.X54.61 Min\_Actual\_Value and Max\_Actual\_Value Test

Purpose: To verify that Min\_Actual\_Value remains less than Max\_Actual\_Value and within the allowable range when either is written to a value that would violate these conditions.

Test Concept: Write a value to Min\_Actual\_Value which is larger than Max\_Actual\_Value. Verify that Max\_Actual\_Value became equal to Min\_Actual\_Value. Next, write a value to Max\_Actual\_Value which is less than Min\_Actual\_Value. Verify that Min\_Actual\_Value became equal to Max\_Actual\_Value.

Verify that neither Min\_Actual\_Value nor Max\_Actual\_Value will accept a value outside the range 1.0 to 100.0.

Configuration Requirements: The IUT shall be configured with a lighting output, O1. Min\_Actual\_Value shall be set to a value less than Max\_Actual\_Value, and Max\_Actual\_Value shall be within the allowable range for Min\_Actual\_Value and not equal to Min\_Actual\_Value's maximum supported value. If the IUT cannot be configured to meet these requirements, then this test shall be skipped.

Test Steps:

- 1. V1 = READ Max\_Actual\_Value
- 2. WRITE Min\_Actual\_Value = V2, a value greater than V1
- 3. VERIFY Max\_Actual\_Value = V2

- 4. WRITE Max\_Actual\_Value = V3, a value less than V2
- 5. VERIFY Min\_Actual\_Value = V3
- 6. TRANSMIT WritePropertyRequest
  - 'Object Identifier' = 01, 'Property Identifier' = Min\_Actual\_Value, 'Property Value' = (any value outside the range 1.0 to 100.0)
- 7. RECEIVE BACnet-Error-PDU, Error Class = PROPERTY, Error Code = VALUE\_OUT\_OF\_RANGE
- TRANSMIT WritePropertyRequest

   'Object Identifier' = 01,
   'Property Identifier' = Max\_Actual\_Value,
   'Property Value' = (any value outside the range 1.0 to 100.0)

   RECEIVE BACnet-Error-PDU,
  - Error Class = PROPERTY, Error Code = VALUE\_OUT\_OF\_RANGE

#### 7.3.2.X54.62 Min\_Actual\_Value and Max\_Actual\_Value Scaling Test

Purpose: To verify that the physical output level changes to the expected scaled value as Present\_Value changes.

Test Concept: Set Min\_Actual\_Value to a value other than the lowest supported minimum value, and set Max\_Actual\_Value to a value other than the highest support value but larger than Min\_Actual\_Value.

Then write 1.0 to Present\_Value and measure the physical output. Repeat the procedure to measure the physical output after writing 100.0 to Present\_Value. After obtaining these upper and lower bound values, write a value between 1.0 and 100.0, measure the physical output, and confirm that the measured value is approximately the same as the expected scaled value.

Configuration Requirements: The IUT shall be configured with a lighting output, O1 that can be observed during the test. O1 shall be configured such that all slots in the Priority\_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1 and Out\_Of\_Service = FALSE.

Test Steps:

- 1. WRITE Min\_Actual\_Value = (a supported value that is not the lowest supported value)
- 2. WRITE Max\_Actual\_Value = (a supported value which is not the highest support value)
- 3. WRITE Present\_Value = 1.0, ARRAY\_INDEX = PTY1
- 4. CHECK(the value of the physical output is Min\_Actual\_Value)
- 5. WRITE Present\_Value = 100.0, ARRAY\_INDEX = PTY1
- 6. CHECK(the value of the physical output is Max\_Actual\_Value)
- 7. WRITE Present\_Value = (V1, a value between 1.0 and 100.0 exclusive), ARRAY\_INDEX = PTY1
- 8. MAKE(measure the value of the physical output and record in MV)
- 9. CHECK (MV ~= Min\_Actual\_Value + (V1 / 100) \* (Max\_Actual\_Value Min\_Actual\_Value))

#### **Changes:**

[In BTL Specified Tests, modify the test 8.2.1, 8.2.2, 8.3.1, 8.3.2 to test against Lighting Output]

# 8.2.1 Change of Value Notification from an Analog Input, Analog Output, *Lighting Output*, and Analog Value, *Large Analog Value, Integer Value, and Positive Integer Value Object* Present\_Value Property

Reason for Change: Add more primitive value objects. Updated description of the 'List of Values' to improve readability. Updated 'Configuration Requirements'.

Purpose: To verify that the IUT can initiate ConfirmedCOVNotification service requests conveying a change of the Present\_Value property of Analog Input, Analog Output, *Lighting Output*, and Analog Value, *Large Analog Value*, *Integer Value*, and Positive Integer Value objects.

Test Concept: A subscription for COV notifications is established, using a Lifetime of L. L shall be set to a value less than 24 hours and large enough to complete the test. The Present\_Value of the monitored object is changed by an amount less than the COV increment and it is verified that no COV notification is received. The Present\_Value is then changed by an amount greater than the COV increment and a notification shall be received. The Present\_Value may be changed using the WriteProperty service or by another means such as changing the input signal represented by an Analog Input object. For some implementations it may be necessary to write to the Out\_Of\_Service property first to accomplish this task. For implementations where it is not possible to write to these properties at all the vendor shall provide an alternative trigger mechanism to accomplish this task. All of these methods are equally acceptable.

Configuration Requirements: At the beginning of the test, the Out\_Of\_Service property shall have a value of FALSE. Select an object where Present\_Value is not expected to change outside the tester's control by more than COV\_Increment or which has a writable Out\_Of\_Service.

Test Steps:

REPEAT X = (one supported object of each type from the set Analog Input, Analog Output, and Analog Value, *Large Analog Value, Integer Value, and Positive Integer Value*) DO {

1	TPANSMIT SubscribeCOV Pequest	
1.	'Salaarihar Draassa Idartifiaa'	(and the base of the TD)
	Subscriber Process Identifier =	(any value $> 0$ chosen by the 1D),
	Monitored Object Identifier =	X,
	Issue Confirmed Notifications =	TRUE,
_	'Lifetime' =	L
2.	RECEIVE BACnet-SimpleACK-PDU	
3.	BEFORE Notification Fail Time	
	RECEIVE ConfirmedCOVNotification-Rec	quest,
	'Subscriber Process Identifier' =	(the same value used in step 1),
	'Initiating Device Identifier' =	IUT,
	'Monitored Object Identifier' =	Х,
	'Time Remaining' =	(any value appropriate for the Lifetime selected),
	'List of Values' =	(the initial Present_Value and initial Status_Flags)
4.	TRANSMIT BACnet-SimpleACK-PDU	
5.	TRANSMIT ReadProperty-Request,	
	'Object Identifier' =	Х,
	'Property Identifier' =	COV_Increment
6.	RECEIVE BACnet-ComplexACK-PDU,	
	'Object Identifier' =	Х,
	'Property Identifier' =	COV_Increment,
	'Property Value' =	(a value "increment" that will be used below)
7.	IF (Out_Of_Service is writable) THEN	
	WRITE X, Out_Of_Service = TRUE	
	<b>BEFORE</b> Notification Fail Time	
	RECEIVE ConfirmedCOVNotification	on-Request,
	'Subscriber Process Identifier' =	(the same value used in step 1),
	'Initiating Device Identifier' =	IUT,
	'Monitored Object Identifier' =	X,
	'Time Remaining' =	(any value appropriate for the Lifetime selected),
	'List of Values' =	(ReportedPV = any value appropriate for the curren
Pre	sent_Value, and new Status_Flags)	
	TRANSMIT BACnet-SimpleACK-PDU	
8.	IF (Present_Value is now writable) THEN	
	/	

WRITE X, Present\_Value = (any value that differs from "initial Present\_Value" *ReportedPV* by less than "increment") ELSE

MAKE (Present\_Value = any value that differs from "initial Present\_Value" *ReportedPV* by less than "increment")

#### 9. WAIT Notification Fail Time

10. CHECK (verify that no COV notification was transmitted)

11. IF (Present\_Value is now writable) THEN

WRITE X, Present\_Value = (any value that differs from <u>"initial Present\_Value"</u> *ReportedPV* by an amount greater than "increment")

#### RECEIVE BACnet-SimpleACK-PDU

#### ELSE

MAKE (Present\_Value = any value that differs from "initial Present\_Value" *ReportedPV* by an amount greater than "increment")

#### 12. BEFORE NotificationFailTime

RECEIVE ConfirmedCOVNotification-Request,

		1
	'Subscriber Process Identifier' =	(the same value used in step 1),
	'Initiating Device Identifier' =	IUT,
	'Monitored Object Identifier' =	Х,
	'Time Remaining' =	(any value appropriate for the Lifetime selected),
	'List of Values' =	(the new Present_Value and new Status_Flags)
13.	TRANSMIT BACnet-SimpleACK-PDU	
14.	TRANSMIT SubscribeCOV-Request,	
	'Subscriber Process Identifier' =	(the same value used in step 1),
	'Monitored Object Identifier' =	Х
15.	RECEIVE BACnet-SimpleACK-PDU	
16.	IF (Out_Of_Service is writable) THEN	
	WRITE X, Out_Of_Service =	FALSE
	RECEIVE BACnet SimpleACK PDU	

# 8.2.2 Change of Value Notification from an Analog Input, Analog Output, *Lighting Output*, and Analog Value, *Large Analog Value, Integer Value, and Positive Integer Value* Object Status\_Flags Property

Reason for Change: Add more primitive value objects. Updated 'Configuration Requirements'. Removed extraneous SimpleACKs after WRITE statements. Updated descriptive text for 'List of Value' property.

Purpose: To verify that the IUT can initiate ConfirmedCOVNotification service requests conveying a change of the Status\_Flags property of Analog Input, Analog Output, *Lighting Output*, and Analog Value, *Large Analog Value, Integer Value, and Positive Integer Value* objects.

Test Concept: A subscription for COV notifications is established, using a Lifetime of L. L shall be set to a value less than 24 hours and large enough to complete the test. The Status\_Flags property of the monitored object is then changed and a notification shall be received. The value of the Status-Flags property can be changed by using the WriteProperty service or by another means. For some implementations writing to the Out\_Of\_Service property will accomplish this task. For implementations where it is not possible to write to Status\_Flags or Out\_Of\_Service or change the Status\_Flags by any other means, this test shall be skipped

Configuration Requirements: At the beginning of the test, the Out\_Of\_Service property shall have a value of FALSE. Select an object where Present\_Value is not expected to change outside the tester's control by more than COV\_Increment or which has a writable Out\_Of\_Service.

Test Steps:

REPEAT X = (one supported object of each type from the set Analog Input, Analog Output, and Analog Value, *Lighting Output, Large Analog Value, Integer Value, and Positive Integer Value*) DO {

1.	TRANSMIT SubscribeCOV-Request,	
	'Subscriber Process Identifier' =	(any value $> 0$ chosen by the TD),
	'Monitored Object Identifier' =	X,
	'Issue Confirmed Notifications' =	TRUE,
	'Lifetime' =	L
2.	RECEIVE BACnet-SimpleACK-PDU	
3.	<b>BEFORE</b> Notification Fail Time	
	RECEIVE ConfirmedCOVNotification-Rec	quest,
	'Subscriber Process Identifier' =	(the same value used in step 1),
	'Initiating Device Identifier' =	IUT,
	'Monitored Object Identifier' =	Х,
	'Time Remaining' =	(any value appropriate for the Lifetime selected),
	'List of Values' =	(the initial Present_Value and initial Status_Flags)
4.	TRANSMIT BACnet-SimpleACK-PDU	
_		
5.	WRITE X, Out_Of_Service = TRUE   WRITE	$\Xi X$ , Status_Flags = (a value that differs from initial Status_Flags)
	MAKE (Status_Flags = any value that differs	from initial Status_Flags)
1.	IF (WriteProperty is used in step 5) THEN	
-	- RECEIVE BACnet SimpleACK PDU	
7.	BEFORE Notification Fail Time	
	RECEIVE ConfirmedCOVNotification-R	lequest,
	Subscriber Process Identifier' =	(the same value used in step 1),
	Initiating Device Identifier =	101, V
	Monitored Object Identifier =	$\mathbf{X}_{\mathbf{y}}$
	lime Remaining =	(any value appropriate for the Lifetime selected),
0	TDANSMIT DACRat Simple ACK DDU	(the mitiative current Present_value and new Status_Plags)
о. 0	TRANSMIT DACHEL-SIMPleACK-PDU	
9.	'Subscriber Process Identifier' -	(the same value used in step 1)
	Monitored Object Identifier' -	(the same value used in step 1),
10	DECEIVE DA Creat Simple A CK DDU	Λ
10.	IE (Out Of Service was abanged in step 5) T	UEN
11.	WPITE V Out Of Service – EALSE	LILAN
	$WINTE A, OULOI_SERVICE - FALSE$ $PECEIVE BACnot Simple ACV DDU$	
	RECEIVE DACHEL SHIIPHEACK PDU	

[In BTL Specified Tests, other referenced non-object specific tests for Blink in section 7.3.1.X, applicable to both Lighting Output or Binary Lighting Output objects, were added in BTL-15.0-2.1.0]

## BTL-TP15.0-6.1.0 Tests for the claim of DS-COVP-B

A device claiming DS-COVP-B at Protocol\_Revision 2 or higher shall comply with the following section.

### **Overview:**

Addendum 135-1995*c* added the SubscribeCOVProperty service. This document makes needed changes in the BTL Test Package to claim the DS-COVP-B BIBB.

These changes adapt and extend some existing tests defined in 135.1.

[In Checklist, add DS-COVP-B just after existing DS-COVP-A]

## **4 Data Sharing**

Data Sharing - Change Of Value Property - B		
	R	Base Requirements

[In BTL Test Plan, add a section for DS-COVP-B]

## 4.19 Data Sharing - Change Of Value Property - B

## **4.19.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL - 9.11.1.1 - Confirmed COV Notifications			
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives	Select parameters for an object and property which supports	
		SubscribeCOVProperty. Repeat for each different datatype for which	
		SubscribeCOVProperty is supported.	
	Testing Hints		
	Notes & Results		
BTL	- 9.11.1.2 - Unconfirmed	COV Notifications	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives	Apply the test to an object and property which supports	
		SubscribeCOVProperty. Repeat for each different datatype for which	
		SubscribeCOVProperty is supported.	
	Testing Hints		
	Notes & Results		
135.1-2013 - 9.11.1.4 - Canceling COV Subscriptions			
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 9.11.1.5 - Canceling E	xpired or Non-Existing Subscriptions	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	BTL - 9.11.1.7 Finite Lifetime Subscriptions		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
l	Test Conditionality	Must be executed.	

-			
	Test Directives		
	Testing Hints		
	Notes & Results		
135.1	-2013 - 9.11.1.8 Updatin	g Existing Subscriptions	
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 9.11.1.9 Client-Supplie	ed COV Increment	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL- 9.11.2.1 - The Monitored Object Does Not Support COV Notification			
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	Must be executed, unless all objects support SubscribeCOVProperty on	
		at least one of its properties.	
	Test Directives	Apply the test to a property in an object that does support COV (on any	
		property).	
	Testing Hints		
	Notes & Results		
BTL-	9.11.2.2 - The Monitore	ed Property Does Not Support COV Notification	
	Test Method		
	Configuration	As per BTL Specified Tests.	
	lest Conditionality	Must be executed, unless all objects support SubscribeCOV Property on	
	Test D'accettes	all properties.	
	lest Directives	Apply the test to a property for which the IUI does not support COV,	
		which is contained in an object that does support COV (on a different	
	Testing Hints	property).	
	Notes & Results		
1		1	

[In BTL Specified Tests, derive modified versions of 5 existing tests in 135.1-2013, for DS-COVP-B]

#### 9.11.1.1 Confirmed COV Notifications

Reason for Change: Remove the allowance for devices which do not support both confirmed and unconfirmed notifications.

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to establish a subscription for confirmed COV notifications. An implementation that supports COV reporting cannot respond with an error for both this test and the test in 9.11.1.2.

Test Steps:

1. TRANSMIT SubscribeCOVProperty-Request,

'Subscriber Process Identifier' = (any valid process identifier), 'Monitored Object Identifier' = (any object supporting COV notifications), 'Issue Confirmed Notifications' = TRUE. (any value > 0 if automatic cancellation is supported, otherwise 0), 'Lifetime' = 'Monitored Property Identifier' = (any valid property supporting COV notifications) 2. RECEIVE BACnet-SimpleACK-PDU 3. BEFORE Notification Fail Time IF (the IUT supports confirmed notifications) THEN - RECEIVE BACnetConfirmedCOVNotification-Request, 'Subscriber Process Identifier' = (the same identifier used in the subscription), 'Initiating Device Identifier' = IUT. 'Monitored Object Identifier' = (the same object used in the subscription), 'Time Remaining' = (any value > 0 if automatic cancellation is supported, otherwise 0), 'List of Values' = (values appropriate to the property subscribed to, and any properties the IUT provides with it, such as Status-Flags) other ELSE **RECEIVE BACnet Error PDU**, Error Class = SERVICES, Error Code = SERVICE REQUEST DENIED | OTHER

## 4. TRANSMIT BACnet-SimpleACK-PDU

**9.11.1.2 Unconfirmed COV Notifications** Reason for Change: Remove the allowance for devices which do not support both confirmed and unconfirmed notifications.

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to establish a subscription for Unconfirmed COV notifications. An implementation that supports COV reporting cannot respond with an error for both this test and the test in 9.11.1.1.

Test Steps:

Ι.	TRANSMIT SubscribeCOVProperty-Request,		
	'Subscriber Process Identifier' = (any valid process identifier),		
	'Monitored Object Identifier' = (any object supporting COV notifications),		
	'Issue Confirmed Notifications' = FALSE,		
	'Lifetime' = $(any value > 0 \text{ if automatic cancellation is supported, otherwise } 0),$		
	'Monitored Property Identifier' = (any valid property supporting COV notifications)		
2.	RECEIVE BACnet-SimpleACK-PDU		
3.	BEFORE Notification Fail Time		
	RECEIVE BACnetUnconfirmedCOVNotification-Request,		
	'Subscriber Process Identifier' = (the same identifier used in the subscription),		
	'Initiating Device Identifier' = IUT,		
	'Monitored Object Identifier' = (the same object used in the subscription),		
	'Time Remaining' = (any value > 0-if automatic cancellation is supported, otherwise 0),		
	'List of Values' = (values appropriate to the property subscribed to, and any other properties		
	the IUT provides with it, such as Status-Flags)		
	ELSE		
	RECEIVE BACnet Error PDU,		
	Error Class = SERVICES.		

#### 9.11.1.5 Canceling Expired or Non-Existing Subscriptions

Reason for change: Added missing verification that the IUT did not send a COV notification, and removed superfluous note to tester.

Error Code =

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to cancel a subscription that no longer exists.

Test Steps:

1.	TRANSMIT SubscribeCOVProperty-Request,	
----	----------------------------------------	--

	'Subscriber Process Identifier' =	(any unused process identifier or an identifier from a previously terminated subscription)
	'Monitored Object Identifier' =	(any unused object or an object from a previously terminated subscription)
terminated	'Monitored Property Identifier' =	(any unused property or a property from a previously
terminateu		subscription)

2. RECEIVE BACnet-SimpleACK-PDU

#### 3. WAIT Notification Fail Time

4. MAKE (a change to the monitored object that would cause a COV notification if there were an active subscription)

5. *CHECK(the IUT did not issue a COV notification)* 

# Notes to Tester: The IUT shall not transmit a COV notification message. An error message is not an acceptable response.

#### 9.11.1.7 Finite Lifetime Subscriptions

Reason for change: Updates description of 'Time Remaining' and adds validation that this value counts down as expected.

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to establish a subscription with a

temporary lifetime. Either confirmed or unconfirmed notifications may be used, but at least one of these options must be

supported by the IUT.

#### Test Steps:

1. TRANSMIT SubscribeCOVProperty-Request,

'Subscriber Process Identifier' = (any valid process identifier),

- 'Monitored Object Identifier' = (any object supporting COV notifications),
- 'Issue Confirmed Notifications' = TRUE | FALSE,

'Lifetime' = (a value between 60 seconds and 300 seconds),

'Monitored Property Identifier' = (any valid property supporting COV notifications)

#### 2. RECEIVE BACnet-SimpleACK-PDU

#### **3. BEFORE Notification Fail Time**

3. IF (the subscription was for confirmed notifications) THEN

**BEFORE** Notification Fail Time

RECEIVE BACnetConfirmedCOVNotification-Request,

- 'Subscriber Process Identifier' = (the same identifier used in the subscription),
- 'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

- 'Time Remaining' = (the requested subscription lifetime A value approximately equal to, but not greater than, the requested subscription lifetime),
- 'List of Values' = (values appropriate to the property subscribed to, and any other
- properties the IUT provides with it, such as Status-Flags)

TRANSMIT BACnet-SimpleACK-PDU

ELSE

**BEFORE** Notification Fail Time

RECEIVE BACnetUnconfirmedCOVNotification-Request,

'Subscriber Process Identifier' = (the same identifier used in the subscription),

'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

'List of Values' = (values appropriate to the property subscribed to, and any other properties the IUT provides with it, such as Status-Flags)

4. MAKE (a change to the monitored object that should causes a COV notification)

5. WAIT a period longer than the resolution of the IUT's COV subscription lifetime timer

#### 5. BEFORE Notification Fail Time

6. IF (the subscription was for confirmed notifications) THEN

**BEFORE** Notification Fail Time

RECEIVE BACnetConfirmedCOVNotification-Request,

'Subscriber Process Identifier' = (the same identifier used in the subscription),

'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

'Time Remaining' = (TR: a value greater than 0 and less than the requested subscription lifetime),

'List of Values' = (values appropriate to the property subscribed to, and any other

properties the IUT provides with it, such as Status-Flags)

#### TRANSMIT BACnet-SimpleACK-PDU

ELSE

#### **BEFORE** Notification Fail Time

RECEIVE BACnetUnconfirmedCOVNotification-Request,

'Subscriber Process Identifier' = (the same identifier used in the subscription), 'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

'Time Remaining' = (TR: a value greater than 0 and less than the requested subscription lifetime).

'List of Values' = (values appropriate to the object type of the monitored object including the changed value that triggered the notification)

7. WAIT a period longer than the resolution of the IUT's COV subscription lifetime timer

### 8. MAKE (a change to the monitored object that causes a COV notification)

9. *IF (the subscription was for confirmed notifications) THEN* 

#### BEFORE Notification Fail Time

RECEIVE BACnetUnconfirmedCOVNotification-Request,

'Subscriber Process Identifier' = (the same identifier used in the subscription), 'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

'Time Remaining' = (a value greater than 0 and less than the TR),

'*List of Values'* = (values appropriate to the object type of the monitored object)

#### ELSE

#### **BEFORE** Notification Fail Time

RECEIVE BACnetUnconfirmedCOVNotification-Request,

'Subscriber Process Identifier' = (the same identifier used in the subscription), 'Initiating Device Identifier' = IUT,

'Monitored Object Identifier' = (the same object used in the subscription),

'Time Remaining' = (a value greater than 0 and less than the TR),

'List of Values' = (values appropriate to the object type of the monitored object including the changed value that triggered the notification)

610. WAIT (the lifetime of the subscription)

*711.* MAKE (a change to the monitored object that would cause a COV notification if there were an active subscription)

12. CHECK (verify that the IUT did not transmit a COV notification message)

#### Notes to Tester: The IUT shall not transmit a COV notification message addressed to the TD after step 6.

#### 9.11.1.9 Client-Supplied COV Increment

Reason for Change: Modify the test to work with all numeric datatypes.

Purpose: To verify that the IUT correctly generates COV notifications when the client supplies the COV increment in the SubscribeCOVProperty request. Either confirmed or unconfirmed notifications may be used but at least one of these options must be supported by the IUT.

Test Concept: A subscription for COV notification is made for a property of *numeric* datatype **REAL**. The subscription request specifies a COV increment. The monitored property is changed by an amount less than the increment, and the TD waits to ensure that the IUT does not generate a notification. The monitored property is changed by an amount slightly more than is required to cause a COV notification, and the TD waits for the notification.

Test Configuration: If the property being subscribed to has a related COV\_Increment property in the object, then the value of the COV\_Increment property should be significantly different than the COV increment provided in the subscription service.

Test Steps:

1.	. TRANSMIT SubscribeCOVProperty-Request,		
	'Subscriber Process Identifier' = (any	valid process identifier),	
	'Monitored Object Identifier' = (any	object supporting COV notifications),	
	'Issue Confirmed Notifications' = TRU	E   FALSE,	
	'Lifetime' = (any	value that will ensure no re-subscription is required to complete the	
test	t),		
	'Monitored Property Identifier' = (any	valid property supporting COV notifications),	
	'COV Increment' = (any	valid increment value)	
2.	RECEIVE BACnet-SimpleACK-PDU		
3.	BEFORE Notification Fail Time		
	IF (the subscription was for confirmed no	tifications) THEN	
	RECEIVE BACnetConfirmedCOVN	lotification-Request,	
	'Subscriber Process Identifier' =	(the same identifier used in the subscription),	
	'Initiating Device Identifier' =	IUT,	
	'Monitored Object Identifier' =	(the same object used in the subscription),	
	'Time Remaining' ~=	(the requested lifetime),	
	'List of Values' =	(values appropriate to the object type of the monitored object	
inc	luding		
		the value of monitored property)	
	TRANSMIT BACnet-SimpleACK-P	DU	
	ELSE		
	RECEIVE BACnetUnconfirmedCO	VNotification-Request,	
	'Subscriber Process Identifier' =	(the same identifier used in the subscription),	
	'Initiating Device Identifier' =	IUT,	
	'Monitored Object Identifier' =	(the same object used in the subscription),	
	'Time Remaining' ~=	(the requested lifetime),	
	'List of Values' =	(values appropriate to the object type of the monitored object	
		including the value of monitored property)	
4.	MAKE (the monitored property change by les	s than the COV increment)	

5. CHECK (verify that the IUT did not transmit a notification message for the monitored property)

6. MAKE (the monitored property change by slightly more than COV Increment less the amount changed in step 5)

#### 7. BEFORE Notification Fail Time

IF (the subscription was for confirmed notifications) THEN

RECEIVE BACnetConfirmedCOVNotification-Request,

'Subscriber Process Identifier' =	(the same identifier used in the subscription),
'Initiating Device Identifier' =	IUT,
'Monitored Object Identifier' =	(the same object used in the subscription),
'Time Remaining' =	?,
'List of Values' =	(values appropriate to the object type of the monitored object
	including the changed value that triggered the notification)

TRANSMIT BACnet-SimpleACK-PDU

#### ELSE

RECEIVE BACnetUnconfirmedCOVN	lotification-Request,
'Subscriber Process Identifier' =	(the same identifier used in the subscription),
'Initiating Device Identifier' =	IUT,
'Monitored Object Identifier' =	(the same object used in the subscription),
'Time Remaining' =	?,
'List of Values' =	(values appropriate to the object type of the monitored object
	including the changed value that triggered the notification)

[In BTL Specified Tests, derive modified versions of two existing tests in 135.1-2013, with specified responses different from those in the modified versions of those tests in 135.1-2013o, for DS-COVP-B]

#### 9.11.2.1 The Monitored Object Does Not Support COV Notification

Reason for Change: Update the accepted error responses as per changes made in Protocol\_Revision 15.

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to establish a subscription when the monitored object does not support COV notifications.

Test Steps:

1.	TRANSMIT Subscribe	COVProperty-Requ	est,
	'Subscriber Process I	dentifier' = (any	valid process identifier).
	'Monitored Object Id	lentifier' = (any ob	viect that does not support COV notifications).
	'Issue Confirmed No	tifications' =	TRUE
	'Lifetime' = 60	tinoutions	11(02),
	'Monitored Property	Identifier' - (any	property in the object)
2	DECEIVE DACnot Error D	$\operatorname{Identifier} = (\operatorname{any})$	property in the object)
z.	Emer Class	<del>JU,</del> SEDVICI	28
	Effor Class =	<u>SEKVICI</u>	
	Error Code =	SERVICE	E_REQUEST_DENIED   OTHER
2.	IF (Protocol_Revision i	s present and Proto	$col_Revision \ge 15$ ) THEN
	RECEIVE		
	(BAC	net-Error-PDU,	
		Error Class =	OBJECT,
		Error Code =	<i>OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED)</i> /
	(BAC	net-Error-PDU,	
		Error Class =	PROPERTY,
		Error Code =	NOT COV PROPERTY)
	ELSE		, <u>,</u> ,
	RECEIVE		
	(BAC	net-Error-PDU,	
		Error Class =	SERVICES,
		Error Code =	SERVICE REQUEST DENIED   OTHER)
	(BAC	net-Error-PDU.	- ~ - ' ' ' '
	(	Error Class =	OBJECT.
		Error Code –	OPTIONAL FUNCTIONALITY NOT SUPPORTED)
		21101 0000 =	of House_Fortenering (of _5011 OKIED)

(BACnet-Error-PDU, Error Class = PROPERTY, Error Code = NOT\_COV\_PROPERTY)

#### 9.11.2.2 The Monitored Property Does Not Support COV Notification

Reason for Change: Update the accepted error responses as per changes made in Protocol\_Revision 15.

Purpose: To verify that the IUT correctly responds to a SubscribeCOVProperty request to establish a subscription when the monitored object supports COV notifications but not on the requested property.

Test Steps:

1.	TRANSMIT SubscribeCOVProperty-Request,
	'Subscriber Process Identifier' = (any valid process identifier),
	'Monitored Object Identifier' = (any object that supports COV notifications),
	'Issue Confirmed Notifications' = TRUE,
	'Lifetime' = 60,
	'Monitored Property Identifier' = (any property that does not support COV notifications)
2.	RECEIVE BACnet Error PDU,
	Error Class = SERVICES,
	Error Code = SERVICE_REQUEST_DENIED   OTHER
2.	IF (Protocol_Revision is present and Protocol_Revision $\geq 15$ ) THEN
	RECEIVE BACnet-Error-PDU,
	Error Class = PROPERTY,
	<i>Error Code</i> = <i>NOT_COV_PROPERTY</i>
	ELSE
	RECEIVE
	(BACnet-Error-PDU,
	Error Class = SERVICES,
	Error Code = SERVICE_REQUEST_DENIED   OTHER)
	(BACnet-Error-PDU,
	Error Class = PROPERTY,
	$Error \ Code = NOT\_COV\_PROPERTY)$

## BTL-TP15.0-7.1.0 Tests for the claim of NM-FDR-A

A device claiming NM-FDR-A at any Protocol\_Revision shall comply with the following section. **Overview:** 

Addendum 135-2012*al* added the NM-FDR-A BIBB. This document makes needed changes in the BTL Test Package to provide for claiming the BIBB.

These changes are not contained in any SSPC proposal.

#### **Changes:**

[In BTL Checklist, add new Network Management - Foreign Device Registration - A section]

### **10** Network Management

Support	Listing	Option	
<b>NT</b> /			
Netv	Network Management - Foreign Device Registration - A		
	R	Base Requirements	
	BTL-R Supports configurable BBMD Address		
	0	Supports a mode where it transmits a Broadcast at Startup	
	O Supports configurable Time-to-Live		

[In BTL Test Plan, add new Network Management - Foreign Device Registration - A sections at end of section 10]

### **10** Network Management

### 10.X2 Network Management - Foreign Device Registration - A

These tests are designed for testing the recurring initiation of a Register-Foreign-Device BVLL to the configured BBMD.

### **10.X2.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

135.1	135.1-2013 - 14.8 - Register-Foreign-Device Test	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed.
	<b>Test Directives</b>	
	<b>Testing Hints</b>	
	Notes & Results	
BTL	BTL - 14.9.X1 - Register-Foreign-Device Enable and Disable Test	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.

	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 14.9.X2 - Recurring l	Register-Foreign-Device Test
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 14.1.6 - Distrib	ute-Broadcast-To-Network
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 14.1.9 - Origina	al-Unicast-NPDU
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	<u>- 14.1.10 - Forwarded-</u>	NPDU (Two-hop Distribution)
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

## **10.X2.2 Supports configurable BBMD Address**

The IUT supports a configurable BBMD Address to which it sends Register-Foreign-Device NPDU.

BTL ·	BTL - 14.9.X3 - BBMD Address Configuration Test	
	Test Method Manual	
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	

Testing Hints	
Notes & Results	

### 10.X2.3 Supports a mode where it transmits a Broadcast at Startup

The IUT transmits a Broadcast at Startup, which can be observed preceded by the sending of Register-Foreign-Device NPDU, when configured as a Foreign Device.

BTL	BTL - 14.9.X4 - Transmits a Broadcast at Startup preceded by Register-Foreign-Device	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

## **10.X2.4 Supports configurable Time-to-Live**

The IUT supports a configurable Time-to-Live which it uses in the Register-Foreign-Device NPDU it sends.

BTL ·	BTL - 14.9.X5 - Time-to-Live Configuration Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	<b>Test Directives</b>		
	Testing Hints		
	Notes & Results		

[Add in BTL Specified tests, these five entirely new tests]

#### 14.9.X1 - Register-Foreign-Device Enable and Disable Test

Reason For Change: This tests that the behavior in test 14.8 can be configured by the product end-user.

Purpose: Verify that the option to issue Register-Foreign-Device requests can be configured by the product end-user.

Test Concept: Using a product end-user interface, configure the mode for use of Register-Foreign-Device requests, and then configure the mode to cease use of Register-Foreign-Device requests.

Configuration Requirements: The means by which the product is here configured shall be part of the product's enduser interface. BBMD1 is the TD simulating a correctly functioning BBMD implementation.

Test Steps:

- 1. MAKE (IUT enter mode for use of Register-Foreign-Device requests)
- 2. RECEIVE DA = BBMD1,
  - Register-Foreign-Device
- 3. TRANSMIT BVLC-Result,
  - 'Result Code' = Successful completion
- 4. MAKE (the IUT not in mode for use of Register-Foreign-Device requests
- 5. WAIT (more than 31 seconds longer than the 'Time-to-Live' parameter used in Register-Foreign-Device requests)
- 6. CHECK (that the IUT did not send any Register-Foreign-Device requests)

#### 14.9.X2 Recurring Register-Foreign-Device Test

Reason For Change: This tests in continuous manner what 14.9.1 tests just once.

Purpose: Verify that mode for use of Register-Foreign-Device repeats the Registration recurringly, when in that mode.

Test Concept: IUT is put in a mode to use Register-Foreign-Device requests, and it is observed that Register-Foreign-Device requests are sent sufficiently frequently to prevent expiration of the registration at the BBMD.

Configuration Requirements: The product's setting of 'BBMD Address' parameter is configured as BBMD1. BBMD1 is the TD simulating a correctly functioning BBMD implementation.

Test Steps:

5.

- 1. MAKE (IUT enter mode for use of Register-Foreign-Device requests)
- 2. RECEIVE DA = BBMD1,
  - Register-Foreign-Device
- 3. TRANSMIT BVLC-Result,
  - 'Result Code' = Successful completion
- 4. BEFORE (the time configured for the 'Time-to-Live' parameter used for Register-Foreign-Device requests) RECEIVE DA = BBMD1,

Register-Foreign-Device

TRANSMIT BVLC-Result,

'Result Code' = Successful completion

6. BEFORE (the time configured for the 'Time-to-Live' parameter used for Register-Foreign-Device requests) RECEIVE DA = BBMD1,

Register-Foreign-Device

7. TRANSMIT BVLC-Result,

'Result Code' = Successful completion

Notes to Tester: There is no need for the recurring request to be sent any more quickly than precisely the 'Time-to-Live' since the standard mandates that the BBMD preserve the registration for 30 seconds past the 'Time-to-Live'.

#### 14.9.X3 BBMD Address Configuration Test

Reason For Change: This tests that the behavior in test 14.8 can be configured by the product end-user.

Purpose: Verify that the parameter in Register-Foreign-Device in test 14.8 can be configured by the product end-user.

Test Concept: Using a product end-user interface, configure the 'BBMD Address' parameter that is used in Register-Foreign-Device requests.

Configuration Requirements: The means by which the product is configured for a 'BBMD Address' can be anything in the product's end-user interface. BBMD1 is the TD simulating a correctly functioning BBMD implementation.

Test Steps:

- 1. MAKE (through the product's end-user interface, the setting of 'BBMD Address' parameter equal BBMD1)
- 2. MAKE (IUT enter mode for use of Register-Foreign-Device requests)
- 3. RECEIVE DA = BBMD1,

Register-Foreign-Device

4. TRANSMIT BVLC-Result,

'Result Code' = Successful completion

#### 14.9.X4 Transmits a Broadcast at Startup preceded by Register-Foreign-Device

Reason For Change: This tests in the specific case of startup, what test 14.9.1 expects to observe during ordinary ongoing operation.

Purpose: Verify that mode for use of Register-Foreign-Device and setting of 'BBMD Address' parameter are persistent across reset, and that the issuance of Register-Foreign-Device precedes the first issuance of any broadcast, when in that mode.

Test Concept: IUT is put in a mode to use Register-Foreign-Device requests, persistently so it will be re-established, then IUT is reset, and the timing of Register-Foreign-Device request to re-establish that precedes the first issuance of any broadcast.

Configuration Requirements: The product's setting of 'BBMD Address' parameter is configured as BBMD1. BBMD1 is the TD simulating a correctly functioning BBMD implementation.

Test Steps:

1. MAKE (IUT enter mode for use of Register-Foreign-Device requests, persistently so it will be re-established after any reset)

- 2. MAKE (IUT reset)
- 3. RECEIVE DA = BBMD1,
  - Register-Foreign-Device
- 4. TRANSMIT BVLC-Result,

'Result Code' = Successful completion

- 5. RECEIVE DA = BBMD1, Distribute-Broadcast-To-Network, NPDU = (any broadcast)
- 6. TRANSMIT BVLC-Result, 'Result Code' = Successful completion

Notes to Tester: For the I-Am, one can precede the Register-Foreign-Device command, as long as then after the Register-Foreign-Device occurs, it is followed by a Distribute-Broadcast-To-Network again, of that I-Am.

#### 14.9.X5 Time-to-Live Configuration Test

Reason For Change: Adds verification that the behavior in test 14.8 can be configured by the product end-user.

Purpose: Verify that the parameter in Register-Foreign-Device in test 14.8 can be configured by the product end-user.

Test Concept: Using a product end-user interface, configure the 'Time-to-Live' parameter that is used in Register-Foreign-Device requests.

Configuration Requirements: The means by which the product is configured can be anything in the product's end-user interface. BBMD1 is the TD simulating a correctly functioning BBMD implementation.

Test Steps:

- 1. MAKE (through the product's end-user interface, the setting of 'Time-to-Live' parameter equal 120)
- 2. MAKE (IUT enter mode for use of Register-Foreign-Device requests)
- 3. RECEIVE DA = BBMD1,

Register-Foreign-Device,

- 'Time-to-Live' = 120 4. TRANSMIT BVLC-Result,
  - 'Result Code' = Successful completion

## BTL-TP15.0-8.1.0 Tests for the claim of GW-EO-B

A device claiming GW-EO-B at any Protocol\_Revision shall comply with the following section.

#### **Overview:**

Addendum 135-2012*al* added the GW-EO-B BIBB definition. This document makes needed changes in the BTL Test Package to claim the GW-EO-B BIBB.

These changes are not contained in any SSPC proposal.

[In BTL Checklist, add two Optional sections, and remove the footnote in Gateway - Embedded Objects - B tests in section 11 2]

### 11 Gateway

Gateway - Embedded Objects - B			
	R <sup>1</sup> Base Requirements		
	0	Supports writes that affect values in "gatewayed" devices	
	O Supports Command Prioritization		
	<sup>4</sup> Contact BTL for interim tests for this BIBB.		

[In BTL Test Plan, add Gateway - Embedded Objects - B tests in section 11 2]

### 11 Gateway

•••

## 11.2 Gateway - Embedded Objects - B

### **11.2.1 Base Requirement**

Base requirements must be met by any IUT that claims GW-EO-B.

BTL - 9.18.1.X8 - ReadProperty gateway object when non-BACnet device offline				
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	Must be executed.		
	Test Directives	The test shall be conducted upon an object which is representing		
		information arriving through a Gateway.		
	Testing Hints			
	Notes & Results			
BTL - 9.20.1.X9 - ReadPropertyMultiple gateway object when non-BACnet device offline				
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	<b>Test Conditionality</b>	If IUT does not support ReadPropertyMultiple service then this test shall be		
		skipped.		
	<b>Test Directives</b>	The test shall be conducted upon an object which is representing		
		information arriving through a Gateway.		
	Testing Hints			

	Notes & Results			
BIL	BTL - 9.21.1.X10 - ReadRange gateway object when non-BACnet device offline			
	Test Method	Manual		
	Configuration	As per BTL Specified Tests.		
	Test Conditionality	If IUT does not support ReadRange service then this test shall be skipped.		
		If IUT support ReadRange service but does not support list property that		
		maps on to non-BACnet devices, this test shall be skipped.		
	Test Directives	The test shall be conducted upon an object which is representing		
		information arriving through a Gateway.		
	Testing Hints			
	Notes & Results			

## 11.2.2 Supports writes that affect values in "gatewayed" devices

The IUT supports DS-WP-B to write values to "gatewayed" devices.

Verify Checklist				
	Test Method	Manual		
	Configuration			
	<b>Test Conditionality</b>	Must be executed.		
	<b>Test Directives</b>	Verify that the IUT claims support for DS-WP-B.		
	<b>Testing Hints</b>			
	Notes & Results			
BTL	- 9.22.1.X11 - WritePro	operty gateway object when non-BACnet device offline		
	Test Method	Manual		
	Configuration	As per <b>BTL Specified Tests.</b>		
	Test Conditionality	Must be executed if an object which is representing information arriving		
		through a Gateway contains any writable properties		
	Test Directives	The test shall be conducted upon an object which is representing		
		information arriving through a Gateway.		
	Testing Hints			
	Notes & Results			
DTI				
BTL - 9.23.1.X12 - WritePropertyMultiple gateway object when non-BACnet device offline				
	Test Method	Manual		
	Configuration	As per <b>BIL Specified Tests</b> .		
	Test Conditionality	If IUT does not support WritePropertyMultiple service then this test shall		
		be skipped. Execute this test if an object which is representing information		
	T ( D' ('	arriving through a Gateway contains any writable properties.		
	Test Directives	The test shall be conducted upon an object which is representing		
	T (* 11* )	information arriving through a Gateway.		
	Testing Hints			
	Notes & Results			

## **11.2.3 Supports Command Prioritization**

135.1-2013 - 7.3.1.2 - Relinquish Default Test				
	Test Method	Manual		
	Configuration	As per ASHRAE 135.1-2013.		
	<b>Test Conditionality</b>	Must be executed.		
	Test Directives	The test shall be conducted upon an object which is representing		
		information arriving through a Gateway. If no object can be made to meet the configuration requirements, this test shall be skipped.		
	Testing Hints			
	Notes & Results			
135.1	135.1-2013 - 7.3.1.3 - Command Prioritization Test			
	Test Method	Manual		
	Configuration	As per ASHRAE 135.1-2013.		
	<b>Test Conditionality</b>	Must be executed.		
	Test Directives	The test shall be conducted upon an object which is representing information arriving through a Gateway.		
	Testing Hints			
	Notes & Results			

Gateways are required to implement Priority\_Array properties correctly with all 16 entries

[In BTL Specified Test add four new tests as shown, each appended to the section for tests of the service.]

#### 9.18.1.X8 ReadProperty service when non-BACnet device offline

Purpose: To verify that ReadProperty Service executes successfully when non-BACnet device is offline or not in communication.

Test Concept: Object1 is an object which is representing information arriving through a Gateway. P1 is a property in Object1.

Test Steps:

- 1. CHECK (any vendor-specified indication, that the non-BACnet device is offline)
- TRANSMIT ReadProperty Request, 'Object Identifier' = Object1, 'Property Identifier' = P1

```
3. (RECEIVE BACnet-Abort-PDU,
```

```
'Abort Reason' = APPLICATION_EXCEEDED_REPLY_TIME) |
```

```
(RECEIVE ReadProperty-ACK,
```

```
'Object Identifier' = Object1,
```

```
'Property Identifier' = P1,
```

```
'Property Value' = (V, any valid value))
```

### 9.20.1.X9 ReadPropertyMultiple gateway object when non-BACnet device offline

Purpose: To verify that ReadPropertyMultiple Service executes successfully and needs only access to the local object, or returns the appropriate error, when the gateway to the non-BACnet device is offline or not in communication.
Test Concept: Object1 is an object which is representing information arriving through a Gateway. P1 is a property in Object1.

Configuration Requirement: The non-BACnet device is not connected to the gateway and the gateway knows that the device is offline.

Test Steps:

- 1. CHECK (any vendor-specified indication, that the non-BACnet device is offline)
- TRANSMIT ReadPropertyMultiple-Request, 'Object Identifier' = Object1, 'Property Identifier' = P1
- 3. (RECEIVE BACnet-Abort-PDU, 'Abort Reason' = APPLICATION\_EXCEEDED\_REPLY\_TIME) | (RECEIVE ReadPropertyMultiple-ACK, 'Object Identifier' = Object1, 'Property Identifier' = P1, 'Property Value' = (any valid value))

#### 9.21.1.X10 ReadRange gateway object when non-BACnet device offline

Purpose: To verify that ReadRange Service executes successfully and needs only access to the local object, or returns the appropriate error, when the gateway to the non-BACnet device is offline or not in communication.

Test Concept: Object1 is an object which is representing information arriving through a Gateway. P1 is a property in Object1.

Configuration Requirement: The non-BACnet device is not connected to the gateway and the gateway knows that the device is offline.

Test Steps:

- 1. CHECK (any vendor-specified indication, that the non-BACnet device is offline)
- TRANSMIT ReadRange-Request, 'Object Identifier' = Object1,

'Property Identifier' = P1

3. (RECEIVE BACnet-Abort-PDU,

'Abort Reason' = APPLICATION EXCEEDED REPLY TIME)

(RECEIVE ReadRange-ACK,

'Object Identifier' = Object1,

'Property Identifier' = P1,

'Property Value' = (any valid value))

#### 9.22.1.X11 WriteProperty gateway object when non-BACnet device offline

Purpose: To verify that WritePropertyMultiple Service executes successfully and needs only access to the local object, or returns the appropriate error, when the gateway to the non-BACnet device is offline or not in communication.

Test Concept: Object1 is an object which is representing information arriving through a Gateway. P1 is a property in Object1.

Configuration Requirement: The non-BACnet device is not connected to the gateway and the gateway knows that the device is offline.

#### Test Steps:

- 1. CHECK (any vendor-specified indication, that the non-BACnet device is offline)
- TRANSMIT WriteProperty-Request, 'Object Identifier' = Object1, 'Property Identifier' = P1, 'Property Value' = (any valid value)
- 3. (RECEIVE BACnet-Abort-PDU, 'Abort Reason' = APPLICATION\_EXCEEDED\_REPLY\_TIME) | (RECEIVE Simple-ACK)

#### 9.23.1.X12 WritePropertyMultiple gateway object when non-BACnet device offline

Purpose: To verify that WritePropertyMultiple Service executes successfully and needs only access to the local object, or returns the appropriate error, when the gateway to the non-BACnet device is offline or not in communication.

Test Concept: Object1 is an object which is representing information arriving through a Gateway. P1 is a property in Object1.

Configuration Requirement: The non-BACnet device is not connected to the gateway and the gateway knows that the device is offline.

Test Steps:

- 1. CHECK (any vendor-specified indication, that the non-BACnet device is offline)
- TRANSMIT WritePropertyMultiple-Request, 'Object Identifier' = Object1, 'Property Identifier' = P1
- 3. (RECEIVE BACnet-Abort-PDU, 'Abort Reason' = APPLICATION\_EXCEEDED\_REPLY\_TIME) | (RECEIVE Simple-ACK)

### **BTL-TP15.0-9.1.0:** Life Safety Point object

Devices claiming support for a Life Safety Point object must claim support for Protocol\_Revision 16 and comply with the following section.

#### **Overview:**

Addendum 135-1995*c* added the Life Safety Point object. This document makes needed changes in the BTL Test Package to claim Life Safety Point object.

These changes are not contained in any SSPC proposal.

[In BTL Checklist, add Life Safety Point object type to Section 3, Objects]

Support	Listing	Option
Life	e Safety Poin	nt Object
	R	Base Requirements
	S	Supports writable Out_Of_Service properties
	0	Supports writable Member_Of property
	0	Contains an object with Reliability_Evaluation_Inhibit Property

[In BTL Test Plan, add Life Safety Point object tests in section 3.X50. In the following addition of new clauses of the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold**. *italic* as shown.]

### 3.X50 Life Safety Point Object

### **3.X50.1 Base Requirements**

Base requirements must be met by any IUT that can contain Life Safety Point objects.

## BTL-TP15.0-9.2.0: Life Safety Zone object

Devices claiming support for a Life Safety Zone object must claim support for Protocol\_Revision 16 and comply with the following section.

#### **Overview:**

Addendum 135-1995*c* added the Life Safety Zone object. This document makes needed changes in the BTL Test Package to claim Life Safety Zone object.

These changes are not contained in any SSPC proposal.

[In BTL Checklist, add Life Safety Zone object type to Section 3, Objects]

Support	Listing	Option
Life	Safety Zon	le Object
	R	Base Requirements
	S	Supports writable Out_Of_Service properties
	0	Supports writable Member_Of property
	0	Contains an object with Reliability_Evaluation_Inhibit Property

[In BTL Test Plan, add Life Safety Zone object tests in section 3.X51. In the following addition of new clauses of the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold**. *italic* as shown.]

### 3.X51 Life Safety Zone Object

### **3.X51.1 Base Requirements**

Base requirements must be met by any IUT that can contain Life Safety Zone objects.

BTL-	BTL-7.3.2.15.X6 – Supports writable Mode property		
	Test Method	Manual	
	Configuration	The test shall be executed using a Life Safety Point and Life Safety Zone	
	-	objects.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	<b>Testing Hints</b>		
	Notes & Results		
BTL-	7.3.2.15.X5 – Support	writable Tracking_Value	
	Test Method	Manual	
	Configuration	The test shall be executed using a Life Safety Point and Life Safety Zone	
		objects.	
	<b>Test Conditionality</b>	If Out Of Service can be made TRUE, this test must be executed.	
	<b>Test Directives</b>		
	Testing Hints		

	Notes & Results	
BTL-	7.3.2.15.X9 – Support	Silenced property
	Test Method	Manual
	Configuration	The test shall be executed using a Life Safety Point and Life Safety Zone
	0	objects.
	<b>Test Conditionality</b>	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL-	7.3.2.15.X7 – Support	Operation_Expected property
	Test Method	Manual
	Configuration	As per BTL Specified Tests. The test shall be executed using a Life Safety
		Point and Life Safety Zone objects.
	Test Conditionality	If IUT is capable of generating event notifications then, it Must be
		executed.
	Test Directives	
	<b>Testing Hints</b>	
	Notes & Results	

### 3.X51.2 Supports writable Out\_Of\_Service properties

The Out\_Of\_Service property in Life Safety objects contained in the IUT are writable.

135.1-	35.1-2013 - 7.3.1.1 - Out_Of_Service, Status_Flags, and Reliability Tests		
	Test Method	Manual	
	Configuration	The test shall be executed using a Life Safety Point and Life Safety Zone	
	-	objects.	
	<b>Test Conditionality</b>	If Out_Of_Service can be made TRUE, this test must be executed.	
	<b>Test Directives</b>		
	<b>Testing Hints</b>		
	Notes & Results		

### 3.X51.3 Support writable Member\_Of property

BTL-	BTL-7.3.2.15.X8 – Support Writable Member_Of property		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	Must be executed.	
	<b>Test Directives</b>		
	<b>Testing Hints</b>		
	Notes & Results		

### 3.X51.4 Contains an object with Reliability\_Evaluation\_Inhibit property

The IUT contains, or can be made to contain, a Reliability\_Evaluation\_Inhibit property that is configurable to a value of TRUE.

BTL	BTL - 7.3.1.X8.1 - Reliability_Evaluation_Inhibit Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	If no object exists in the IUT for which fault conditions can be	
		generated, then this test shall be skipped.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 7.3.1.X8.2 - Reliability	_Evaluation_Inhibit Summarization Test	
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	Test Conditionality	If no object exists in the IUT for which fault conditions can be	
		generated, then this test shall be skipped.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### BTL-TP15.0-9.3.0 Tests for the claim of AE-LS-A

A device claiming AE-LS-A at Protocol\_Revision 2 or higher shall comply with the following section.

#### **Overview:**

Addendum 135-1995*c* added the SubscribeCOVProperty service. This document makes needed changes in the BTL Test Package to claim the AE-LS-A BIBB.

These changes adapt and extend some existing tests defined in 135.1.

R	Base Requirements
R	Initiates LifeSafetyOperation requests
R	Executes ConfirmedEventNotifications
R	Executes UnconfirmedEventNotifications
R	Processes intrinsically generated notifications
R	Processes algorithmically generated notifications
R	Processes event notifications with timestamps of the BACnetDateTime form
R	Processes event notifications with timestamps of the Time form
R	Processes event notifications with timestamps of the Sequence Number form
R	Supports AE-ACK-A
R	Supports AE-AS-A

#### Alarm and Event Management - Life Safety - A

[In BTL Test Plan, add Alarm and Event Management - Life Safety - A in section 5.22. In the following addition of new clauses of the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold-italic** as shown.]

### 5.22 Alarm and Event Management - Life Safety - A

### **5.22.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL	BTL - 9.4.X1 - Unsupported Message Text Character Set ConfirmedEventNotification Test		
	Test Method	Manual	
	Configuration	As per BTL Specified Tests.	
	<b>Test Conditionality</b>	If the IUT supports all character sets, this test shall be skipped.	
	Test Directives		
	Testing Hints		
	Notes & Results		
BTL	- 9.5.X1 - Unsupported I	Message Text Character Set UnconfirmedEventNotification Test	
BTL	- 9.5.X1 - Unsupported I Test Method	Message Text Character Set UnconfirmedEventNotification Test Manual	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration	Message Text Character Set UnconfirmedEventNotification Test Manual As per <i>BTL Specified Tests</i> .	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration Test Conditionality	Message Text Character Set UnconfirmedEventNotification Test         Manual         As per BTL Specified Tests.         If the IUT supports all character sets, this test shall be skipped.	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration Test Conditionality Test Directives	Message Text Character Set UnconfirmedEventNotification Test         Manual         As per BTL Specified Tests.         If the IUT supports all character sets, this test shall be skipped.	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration Test Conditionality Test Directives Testing Hints	Message Text Character Set UnconfirmedEventNotification Test         Manual         As per BTL Specified Tests.         If the IUT supports all character sets, this test shall be skipped.	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	Message Text Character Set UnconfirmedEventNotification Test         Manual         As per BTL Specified Tests.         If the IUT supports all character sets, this test shall be skipped.	
BTL	- 9.5.X1 - Unsupported I Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	Message Text Character Set UnconfirmedEventNotification Test         Manual         As per BTL Specified Tests.         If the IUT supports all character sets, this test shall be skipped.	

### 5.22.2 Initiates LifeSafetyOperation requests

135.1-2013 - 8.9.1 - LifeSafetyOperation Service Initiation Tests to an Object		
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1-2013 - 8.9.2 - LifeSafetyOperation Service Initiation Tests to all Objects in a Device		
135.1	<u>-2013 - 8.9.2 - LifeSafet</u>	yOperation Service Initiation Tests to all Objects in a Device
135.1	-2013 - 8.9.2 - LifeSafet Test Method	yOperation Service Initiation Tests to all Objects in a Device Manual
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration	yOperation Service Initiation Tests to all Objects in a Device Manual As per ASHRAE 135.1-2013.
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration Test Conditionality	yOperation Service Initiation Tests to all Objects in a Device Manual As per <i>ASHRAE 135.1-2013</i> . Must be executed.
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration Test Conditionality Test Directives	yOperation Service Initiation Tests to all Objects in a Device Manual As per ASHRAE 135.1-2013. Must be executed.
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration Test Conditionality Test Directives Testing Hints	yOperation Service Initiation Tests to all Objects in a Device Manual As per ASHRAE 135.1-2013. Must be executed.
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	yOperation Service Initiation Tests to all Objects in a Device Manual As per ASHRAE 135.1-2013. Must be executed.
135.1	-2013 - 8.9.2 - LifeSafet Test Method Configuration Test Conditionality Test Directives Testing Hints Notes & Results	yOperation Service Initiation Tests to all Objects in a Device Manual As per ASHRAE 135.1-2013. Must be executed.

### 5.22.3 Executes ConfirmedEventNotifications

The IUT is capable of executing ConfirmedEventNotifications with an Event Type of CHANGE OF LIFE SAFETY. This functionality will be covered by the testing of the individual algorithms.

No Sj	No Specific Test		
	Test Method		
	Configuration		
	Test Conditionality	Must be executed.	
	Test Directives	Verify that the IUT's EPICS claims that it supports the	
		ConfirmedEventNotification service.	
	Testing Hints		
	Notes & Results		

### **5.22.4 Executes UnconfirmedEventNotifications**

The IUT is capable of executing UnconfirmedEventNotifications with an Event Type of CHANGE\_OF\_LIFE\_SAFETY. There are currently no tests defined for this functional item.

No Sj	No Specific Test		
	Test Method		
	Configuration		
	Test Conditionality	Must be executed.	
	Test Directives	Verify that the IUT's EPICS claims that it supports the	
		UnconfirmedEventNotification service.	
	<b>Testing Hints</b>		
	Notes & Results		

### 5.22.5 Processes Intrinsically Generated Notifications

The IUT is capable of executing ConfirmedEventNotifications with an Event Type of CHANGE OF LIFE SAFETY that reference an object type other than Event Enrollment. 135.1-2013 - 9.4.1 - ConfirmedEventNotification Using the Time Form of the 'Timestamp' Parameter and Conveying a Text Message, 135.1-2013 - 9.4.2 - ConfirmedEventNotification Using the DateTime Form of the 'Timestamp' Parameter and no Text Message, or

135.1-2013 - 9.4.3 - ConfirmedEventNotification Using the Sequence Number Form of the 'Timestamp' Parameter and no Text Message

Test Method	Manual
Configuration	As per ASHRAE 135.1-2013.
Test Conditionality	At least one of the tests must be executed with the Event Object Identifier referencing a BACnet object other than an Event Enrollment object.
Test Directives	
Testing Hints	
Notes & Results	

### 5.22.6 Processes Algorithmically Generated Notifications

The IUT is capable of executing ConfirmedEventNotifications with an Event Type of CHANGE\_OF\_LIFE\_SAFETY that reference an Event Enrollment object.

135.1	135.1-2013 - 9.4.1 - ConfirmedEventNotification Using the Time Form of the 'Timestamp'		
Para	Parameter and Conveying a Text Message,		
135.1	-2013 - 9.4.2 - Confirme	dEventNotification Using the DateTime Form of the 'Timestamp'	
Para	meter and no Text Messa	age, or	
135.1	135.1-2013 - 9.4.3 - ConfirmedEventNotification Using the Sequence Number Form of the		
'Timestamp' Parameter and no Text Message			
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	Test Conditionality	At least one of the tests must be executed with the Event Object	
		Identifier referencing an Event Enrollment object.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 5.22.7 Processes Event Notifications with Timestamps of the BACnetDateTime Form

The IUT is capable of executing ConfirmedEventNotifications that contain a timestamp of the BACnetDateTime form.

135.1	135.1-2013 - 9.4.2 - ConfirmedEventNotification Using the DateTime Form of the 'Timestamp'		
Parai	Parameter and no Text Message		
	Test Method	Manual	
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

### 5.22.8 Processes Event Notifications with Timestamps of the Time Form

135.1	135.1-2013 - 9.4.1 - ConfirmedEventNotification Using the Time Form of the 'Timestamp'		
Para	meter and Conveying a [	Text Message	
	Test Method Manual		
	Configuration	As per ASHRAE 135.1-2013.	
	<b>Test Conditionality</b>	Must be executed.	
	Test Directives		
	Testing Hints		
	Notes & Results		

The IUT is capable of executing ConfirmedEventNotifications that contain a timestamp of the Time form.

### 5.22.9 Processes Event Notifications with Timestamps of the Sequence Number Form

The IUT is capable of executing ConfirmedEventNotifications that contain a timestamp of the Sequence Number form.

135.1-2013 - 9.4.3 - ConfirmedEventNotification Using the Sequence Number Form of the		
'Time	stamp' Parameter and	no Text Message
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	<b>Test Conditionality</b>	Must be executed.
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### 5.22.10 Supports AE-ACK-A

The IUT must support AE-ACK-A if it claims support for AE-LS-A.

Verify	Verify Checklist		
	Test Method	Manual	
	Configuration		
	Test Conditionality	Must be executed.	
	Test Directives	Verify that the IUT claims support for AE-ACK-A in the Checklist.	
	Testing Hints		
	Notes & Results		

### 5.22.11 Supports AE-AS-A

The IUT must support AE-AS-A if it claims support for AE-LS-A.

Verify Checklist		
	Test Method	Manual
	Configuration	
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for AE-AS-A in the Checklist.
	Testing Hints	
	Notes & Results	

### BTL-TP15.0-9.4.0 Tests for the claim of AE-LS-B

A device claiming AE-LS-B at Protocol\_Revision 2 or higher shall comply with the following section.

#### **Overview:**

Addendum 135-1995*c* added the LifeSafetyOperation service. This document makes needed changes in the BTL Test Package to claim the AE-LS-B BIBB.

These changes adapt and extend some existing tests defined in 135.1.

[In BTL Test Plan, add Alarm and Event Management - Life Safety - A in section 5.23. In the following addition of the Test Plan, these are indicated as entirely new sections verbatim, with plain text, verbatim **bold**, or verbatim **bold**. *italic* as shown.]

### 5 Alarm and Event Management BIBBs

#### Alarm and Event Management - Life Safety - B

R	Base Requirements	
R	Supports the Notification Class Object	
R	Supports AE-INFO-B	
C1	Implements intrinsic alarming in a Life Safety object	
$C^1$	Supports the CHANGE_OF_LIFE_SAFETY algorithm in Event_Parameters	
$C^2$	Supports AE-ACK-B	
C <sup>3</sup>	Generates event notifications with timestamps of the BACnetDateTime form	
C <sup>3</sup>	Generates event notifications with timestamps of the Sequence Number form	
0	Mode Transition Tests when Event State is Maintained	
0	Supports Event_Message_Texts property	
0	Supports Event_Message_Texts_Config property	
<sup>1</sup> At least one of these options must be supported to claim support for this BIBB.		
<sup>2</sup> Required if EventNotifications with service parameter AckRequired = True can be issued.		
<sup>3</sup> At least one of these options must be supported to claim support for this BIBB. The		
BACnetDateTime form of the timestamp is the recommended option.		

### 5.23 Alarm and Event Management - Life Safety - B

### **5.23.1 Base Requirements**

Base requirements must be met by any IUT claiming conformance to this BIBB.

BTL - 7.3.1.10.1 - Event_Enable Tests for TO_OFFNORMAL and TO_NORMAL		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If the IUT cannot be configured to meet the configuration requirements
		then this test shall be skipped.
	<b>Test Directives</b>	If Event Enrollment objects are supported, ensure this functionality is
		tested on Event Enrollment objects.
	Testing Hints	The BTL will apply this to a single object. The pretester should apply it
		to all objects that support alarm generation.
	Notes & Results	

BTL - 7.3.1.10.1 - Event\_Enable Tests for TO\_OFFNORMAL and TO\_NORMAL

135.1	-2013 - 7.3.1.12 - Notify	_Type Test
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	If the IUT cannot be configured to meet the 135.1-2013 configuration
		requirements then this test shall be skipped.
	Test Directives	If Event Enrollment objects are supported, ensure this functionality is
		tested on Event Enrollment objects.
	Testing Hints	
	Notes & Results	
135.1	-2013 - 8.4.8 - CHANGI	E_OF_LIFE_SAFETY Tests
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
		Any of the 8.4.8 tests can be used to ensure that the IUT properly
		generates ConfirmedEventNotification requests. The specific tests that
		can be executed are detailed under the test cases for the specific
		algorithms. As long as one of the tests is executed using
		ConfirmedEventNotifications, then this test case shall be satisfied.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 8.5.8 - CHANGI	E_OF_LIFE_SAFETY TESTS
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
		Any of the 8.5.8 tests can be used to ensure that the IUT properly
		generates UnconfirmedEventNotification requests. The specific tests that
		can be executed are detailed under the test cases for the specific
		algorithms. As long as one of the tests is executed using
		UnconfirmedEventNotifications, then this test case shall be satisfied.
	Test Directives	
	Testing Hints	
	Notes & Results	
BTL	- 7.3.1.X9.1 - Event_Det	tection_Enable Inhibits Event Generation
	Test Method	Manual
	Configuration	As per <i>BTL Specified Tests</i> .
	Test Conditionality	If Protocol_Revision < 13, then this test shall be skipped.
	Test Directives	The objects selected by the tester should include all variants that differ
		in the set of supported alarming properties, or the writability of any of
		those properties. At least one instance of each variant should be
		L selected
		Selected.
	Testing Hints	
	Testing Hints Notes & Results	
	Testing Hints Notes & Results	
	Testing Hints Notes & Results	
BTL	Testing Hints Notes & Results - 7.3.1.X9.2 - Event_Det	tection_Enable Inhibits FAULT
BTL	Testing Hints Notes & Results - 7.3.1.X9.2 - Event_Det Test Method	tection_Enable Inhibits FAULT Manual

1		
	Test Conditionality	If $Protocol_Revision < 13$ , then this test shall be skipped.
	Test Directives	The objects selected by the tester should include all variants that differ
		in the set of supported alarming properties, or the writability of any of
		those properties. At least one instance of each variant should be
		selected.
	Testing Hints	
	Notes & Results	
BTL	- 7.3.1.X6.1 - Event_Alg	orithm_Inhibit Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If the IUT has no object in which the Event_Algorithm_Inhibit property
		is present and does not support the Event_Algorithm_Inhibit_Ref
		property, or has no object in which Event_Detection_Enable can be
		made TRUE, this test shall be skipped. If the IUT cannot be
		configured to contain any object capable of an event transition, then
		this test shall be skipped.
	Test Directives	The object types selected by the tester should include all variants that
		differ in the set of supported alarming properties, or the writability of
		any of those properties. At least one instance of each variant should be
		selected.
	Testing Hints	
	Notes & Results	
BTL - 7.3.1.X7.1 - Event_Algorithm_Inhibit_Ref Test		
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	If the IUT has no object in which the Event_Algorithm_Inhibit_Ref
		property is present or has no object in which Event_Detection_Enable
		can be made TRUE, this test shall be skipped.
	Test Directives	The object types selected by the tester should include all variants that
		differ in the set of supported alarming properties, or the writability of
		any of those properties. At least one instance of each variant should be
		selected.
	Testing Hints	
	Notes & Results	
BTL	- 7.3.1.X7.2 - Event_Alg	orithm_Inhibit Writable Test
	Test Method	Manual
	Configuration	As per <b>BTL Specified Tests</b> .
	Test Conditionality	If the IUT has no object in which the Event_Algorithm_Inhibit_Ref
		property is absent or can be made uninitialized or has no object in which
		Event_Detection_Enable can be made TRUE, this test shall be skipped.
	Test Directives	The object types selected by the tester should include all variants that
		differ in the set of supported alarming properties, or the writability of
		any of those properties. At least one instance of each variant should be
		selected.
	Testing Hints	
	Notes & Results	
135.1	-2013 - 9.9.1 - Reset Sing	gle Object Execution Tests

	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 9.9.2 - Reset Mu	tiple Object Execution Test
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	
135.1	-2013 - 9.9.3 - Silencing	Execution Test
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
	Notes & Results	

### **5.23.2 Supports the Notification Class Object**

The IUT supports the Notification Class object in order to send notifications.

Verif	y Checklist	
	Test Method	Manual
	Configuration	
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for the Notification Class Object in
		the Checklist.
	Testing Hints	
	Notes & Results	

### **5.23.3 Supports AE-INFO-B**

The IUT must support AE-INFO-B if it claims support for AE-N-I-B.

Verif	Verify Checklist		
	Test Method	Manual	
	Configuration		
	<b>Test Conditionality</b>	Must be executed.	
	<b>Test Directives</b>	Verify that the IUT claims support for AE-INFO-B in the Checklist.	
	Testing Hints		
	Notes & Results		

### 5.23.4 Implements Intrinsic Alarming in a Life Safety object

The IUT contains, or can be made to contain, an object other than an Event Enrollment object that can generate CHANGE\_OF\_LIFE\_SAFETY ConfirmedEventNotifications and UnconfirmedEventNotifications.

Verif	y Checklist	
	Test Method	Manual
	Configuration	
	Test Conditionality	
	Test Directives	This functionality will be tested by the clause 8.4.8 or 8.5.8 tests in that section.
	Testing Hints	
	Notes & Results	

## **5.23.5** Supports the CHANGE\_OF\_LIFE\_SAFETY algorithm in Event\_Parameters

The IUT contains, or can be made to contain an Event Enrollment object that can generate CHANGE\_OF\_LIFE\_SAFETY ConfirmedEventNotifications and UnconfirmedEventNotifications.

Verif	Verify Checklist		
	Test Method	Manual	
	Configuration		
	Test Conditionality		
	Test Directives	Ensure this functionality is tested on Event Enrollment objects by the	
		clause 8.4.8 or 8.5.8 tests in that section.	
	Testing Hints		
	Notes & Results		

### 5.23.6 Supports AE-ACK-B

The IUT supports AE-ACK-B in order to execute the AcknowledgeAlarm Service if the IUT is able to send initiates EventNotifications with service parameter AckRequired = True.

Verif	y Checklist	
	Test Method	Manual
	Configuration	
	Test Conditionality	
	Test Directives	If the IUT cannot be configured to contain any object with an unacknowledged event, then this test shall be skipped.
	Testing Hints	
	Notes & Results	

## **5.23.7** Generates Event Notifications with Timestamps of the BACnetDateTime Form

The IUT generates, or can be made to generate, ConfirmedEventNotifications with the Time Stamp parameter taking the BACnetDateTime form.

135.1-2013 - 8.4.8 - CHANGE_OF_LIFE_SAFETY Tests		
	Test Method	Manual

Configuration	As per ASHRAE 135.1-2013.
Test Conditionality	If the IUT supports AE-N-I-B, these tests may be skipped.
	Any of the 8.4.8 or 8.5.8 tests can be used to ensure that the IUT
	properly generates ConfirmedEventNotification requests using the
	BACnetDateTime form. The specific tests that can be executed are
	detailed under the test cases for the specific algorithms. As long as one
	of the tests is executed using ConfirmedEventNotifications and the
	notification that is generated contains a timestamp of the
	BACnetDateTime form, then this test case shall be satisfied.
Test Directives	
Testing Hints	
Notes & Results	

# **5.23.8** Generates Event Notifications with Timestamps of the Sequence Number Form

The IUT generates, or can be made to generate, ConfirmedEventNotifications with the Time Stamp parameter taking the Sequence Number form.

135.1	5.1-2013 - 8.4.8 - CHANGE_OF_LIFE_SAFETY Tests	
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	If the IUT supports AE-N-I-B, these tests may be skipped. Any of the 8.4.8 or 8.5.8 tests can be used to ensure that the IUT properly generates ConfirmedEventNotification requests using the Sequence Number form. The specific tests that can be executed are detailed under the test cases for the specific algorithms. As long as one of the tests is executed using ConfirmedEventNotifications and the notification that is generated contains a timestamp of the Sequence
	Test Directives	
	Testing Hints	
	Notes & Results	

### 5.23.9 Mode Transition Tests when Event State is Maintained

135.1-2013 - 8.4.8.7 - Mode Transition Tests when Event State is Maintained		
	Test Method	Manual
	Configuration	As per ASHRAE 135.1-2013.
	Test Conditionality	Must be executed
	<b>Test Directives</b>	
	Testing Hints	
	Notes & Results	

### 5.23.10 Supports Event\_Message\_Texts property

The IUT contains one or more objects that support the Event\_Message\_Texts property.

### BTL - 7.3.1.X4 - Event\_Message\_Texts Tests

Test Method

Configuration	As per BTL Specified Tests.
Test Conditionality	Must be executed.
Test Directives	Repeat test once for each object type in the IUT that contains an
	Event_Message_Texts property.
Testing Hints	
Notes & Results	
Notes & Results	

### 5.23.11 Supports Event\_Message\_Texts\_Config property

The IUT contains one or more objects that support the Event\_Message\_Texts\_Config property.

BTL	BTL - 7.3.1.X5 - Event_Message_Texts_Config Test	
	Test Method	Manual
	Configuration	As per BTL Specified Tests.
	Test Conditionality	Must be executed.
	Test Directives	Repeat for each supported transition type (TO_OFFNORMAL,
		TO_FAULT, TO_NORMAL). Different objects may be selected for
		different transitions.
	Testing Hints	
	Notes & Results	

[In BTL Specified Tests, add five new tests 7.3.2.15.X5 through 7.3.2.15.X9 as indicated.]

#### 7.3.2.15.X5 Writable Tracking\_Value

Purpose: This test case verifies that Present\_Value equals Tracking\_Value, when Tracking\_Value is writable.

Test Concept: It verifies the interrelationship between the Tracking\_Value, Status Flags and Present\_Value properties. This test applies to Life Safety Zone and Life Safety point object. The tester will select one instance of each appropriate object type and test it as described.

Configuration Requirements: The test shall start with Event\_State equal to NORMAL. If writing to the Tracking\_Value is only possible while Out\_Of\_Service equals TRUE, then the test shall start with Out\_Of\_Service equal to TRUE. If the Out\_Of\_Service property of the object under test is not writable, and the value of the Tracking\_Value property cannot be changed by other means, then this test shall be omitted.

Test Steps:

- 1. VERIFY Event\_State = Normal
- 2. WRITE Tracking\_Value = X (any value that corresponds to an Event\_State of NORMAL)
- 3. VERIFY Tracking\_Value = X
- 4. VERIFY Present\_Value = X

#### 7.3.2.15.X6 Supports Writable Mode property

Purpose: To verify that the Mode property takes one of the values found in the Accepted\_Modes property.

Test Concept: It verifies the interrelationship between the Mode, and Accepted\_Modes properties. This test applies to Life Safety Zone and Life Safety point object. The tester will select one instance of each appropriate object type and test it as described.

Test Steps:

READ AM = Accepted\_Modes
 TRANSMIT WriteProperty-Request

	'Object Identifier' =	(the object being tested),
	'Property Identifier' =	Mode,
	'Property Value' =	X (Any valid value from list of AM)
3.	RECEIVE SimpleACK-PDU	
4.	VERIFY Mode = $X$	
5.	TRANSMIT WriteProperty-R	equest
	'Object Identifier' =	(the object being tested),
	'Property Identifier'=	Mode,
	'Property Value' =	X (Any invalid value, which is not present in AM)
6.	RECEIVE BACnet-Error-PD	U,
	Error Class = PROPE	RTY,
	Error Code = VALUE	_OUT_OF_RANGE

#### 7.3.2.15.X7 Support Operation\_Expected Property

Purpose: To verify that the Operation\_Expected property takes on the value of ConfirmedEventNotification-Request.

Test Concept: It verifies the interrelationship between the Operation\_Expected property, and ConfirmedEventNotification-Request. This test applies to Life Safety Zone and Life Safety point object. The IUT will select one instance of each appropriate object type and test it as described.

Test Steps:

- 1. MAKE (the IUT send an ConfirmedEventNotification)
- 2. RECEIVE ConfirmedEventNotification-Request,

	A
'Process Identifier' =	(any valid process identifier),
'Initiating Device Identifier' =	TD,
'Event Object Identifier' =	(any Life-Safety object),
'Time Stamp' =	(the current local time),
'Notification Class' =	(any valid notification class),
'Priority' =	(any valid priority),
'Event Type' =	CHANGE-OF-LIFE-SAFETY,
'Message Text' =	(any character string),
'Notify Type' =	ALARM   EVENT,
'AckRequired' =	TRUE  FALSE,
'From State' =	NORMAL,
'To State' =	(any non-normal state appropriate to the event type),
'Event Values' =	( New State: (Any Valid State), New-Mode: (Any Valid Mode),
	Status-Flag: (TRUE, FALSE, ?, ?), Operation_Expected: ("X ", Any
	Valid operation))
	'Process Identifier' = 'Initiating Device Identifier' = 'Event Object Identifier' = 'Time Stamp' = 'Notification Class' = 'Priority' = 'Event Type' = 'Message Text' = 'Message Text' = 'Notify Type' = 'AckRequired' = 'From State' = 'To State' = 'Event Values' =

3. VERIFY Operation\_Expected = X ( operation expected in the step 2)

#### 7.3.2.15.X8 Support Writable Member\_Of property

Purpose: To verify that the Member\_Of property takes only supported values of the Life Safety objects within the IUT.

Test Concept: If the property is writable and is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result (-), if the property is not writable and if the value of the property cannot be changed by other means, then this test shall be omitted. The tester will select one instance of each appropriate object type and test it as described.

Test Steps:

1.	TRANSMIT WriteProperty-Request,		
	Object Identifier' =	(life safety object),	
	'Property Identifier' =	Member_Of	
	'Property Value' =	X (any valid life safety object)	
2.	<b>RECEIVE Simple-ACK-PD</b>	U,	
3.	TRANSMIT ReadProperty-Request,		
	'Object Identifier' =	(life safety object),	
	'Property Identifier' =	Member_Of	

4.	. RECEIVE ReadProperty-ACK,		
	'Object Identifier' =	(the object being tested),	
	'Property Identifier' =	Member_Of	
	'Property Value' =	X (the value used in step 1)	

#### 7.3.2.15.X9 Silenced Property test

Purpose: This test verifies the behavior of Silenced property.

Test Concept: Verify the interrelationship between the Silenced property and any audible or visual indication that has been silenced by the receipt of a LifeSafetyOperation service request or a local process. If the Silenced property of the object under test is unchanging by means of a LifeSafetyOperation service requests, because none of the silencing operations are supported, then this test shall be omitted. This test applies to Life Safety Zone and Life Safety Point object. The tester will select one instance of each appropriate object type and test it as described.

Test Steps:

- 1. READ InitialSilencedState = Silenced
- 2. TRANSMIT LifeSafetyOperation-Request,
  - 'Requesting Process Identifier' = (any valid identifier),
    - 'Requesting Source' = (any valid character string),
    - 'Request' = (any supported LifeSafetyOperation request transmitted to silence the sounder/strobe),

'Object Identifier' = (the selected object)

- 3. RECEIVE BACnet-SimpleACK-PDU
- 4. CHECK (Sounder/Strobe inactive)
- 5. READ ResultingSilencedState = Silenced
- 6. CHECK (the ResultingSilencedState is equal to the InitialSilencedState, modified by the LifeSafetyOperation request transmitted)

## BTL-TP15.1-0.1.0 File object

A device claiming File object at Protocol\_Revision 2 or higher shall comply with the following section. Rationale: The uses of File for a purpose other than Backup and Restore, all seem to be essentially proprietary.

[In BTL Checklist, add File object type to Section 3, Objects]

Support	Listing	Option	
•••			
File			
	R	Base Requirements	
	$C^1$	Supports DM-BR-B	
	C <sup>1</sup>	Supports a File object for a purpose other than Backup and Restore	
	0	Contains a writable File for a purpose other than Backup and Restore	
	<sup>1</sup> At least one of these options is required if the IUT supports the File object type.		

[In BTL Test Plan, add File object tests in section 3.X48]

### **3.X48 File**

### **3.X48.1 Base Requirements**

For File object, there are no base requirements.

### 3.X48.2 Supports DM-BR-B

The IUT supports a data File that is readable and writable during Backup and Restore using AtomicReadFile and AtomicWriteFile requests.

Verify Checklist		
	Test Method	Manual
	Configuration	
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for Device Management - Backup
		and Restore - B in the Checklist.
	Testing Hints	
	Notes & Results	

### 3.X48.3 Supports a File object for a purpose other than Backup and Restore

For a device which contains a File object for a purpose other than Backup and Restore, there are no testing requirements.

### 3.X48.4 Contains a writable File for a purpose other than Backup and Restore

For a device which contains a writable File object for a purpose other than Backup and Restore, there are no testing requirements.